Hyperspectral Mineral Identification using SVM and SOM

Arash Iranzad

Department of Computer Science

Submitted in partial fulfillment
of the requirements for the degree of

Master of Science

Faculty of Mathematics and Science, Brock University
St. Catharines, Ontario

To my parents and my brother, who taught me how to think and how to live.

Abstract

Remote sensing techniques involving hyperspectral imagery have applications in a number of sciences that study some aspects of the surface of the planet. The analysis of hyperspectral images is complex because of the large amount of information involved and the noise within that data. Investigating images with regard to identify minerals, rocks, vegetation and other materials is an application of hyperspectral remote sensing in the earth sciences. This thesis evaluates the performance of two classification and clustering techniques on hyperspectral images for mineral identification. Support Vector Machines (SVM) and Self-Organizing Maps (SOM) are applied as classification and clustering techniques, respectively. Principal Component Analysis (PCA) is used to prepare the data to be analyzed. The purpose of using PCA is to reduce the amount of data that needs to be processed by identifying the most important components within the data. A well-studied dataset from Cuprite, Nevada and a dataset of more complex data from Baffin Island were used to assess the performance of these techniques. The main goal of this research study is to evaluate the advantage of training a classifier based on a small amount of data compared to an unsupervised method. Determining the effect of feature extraction on the accuracy of the clustering and classification method is another goal of this research. This thesis concludes that using PCA increases the learning accuracy, and especially so in classification. SVM classifies Cuprite data with a high precision and the SOM challenges SVM on datasets with high level of noise (like Baffin Island).

Acknowledgements

First and foremost I would like to express my deepest gratitude towards my supervisor Prof. Frank Fueten for his support and assistance throughout the course of my research and this thesis. His kind words and insightful suggestions have always guided me in the right direction when I was lost in the sea of papers.
In addition, I would like to thank my co-supervisor Prof. Beatrice Ombuki-Berman for her invaluable guidance. Also Prof. Brian Ross for his constant help during my study. Furthermore, my friends Farzan, Mehran, Farhad, Mahdi, Koosha, Manas, Yousef, Reza and others.
Last but not least I would like to thank my be loved family, for their constant support through all I have ever done. Also, my girlfriend, Fatemeh, for her patience and support during my studies.

A.I

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Remote sensing refers to the technique of obtaining information about a remote location without any physical contact with it [74]. Some remote sensing processes utilize several images taken from the same target with different sensors which have different sensitivity levels. Each of these sensors records a narrow band of spectra. Hyperspectral imagery is a relatively new technology which greatly aids remote sensing [50]. The surface of almost every terrain on Earth could theoretically be mapped by hyperspectral images. There are, however, some complications associated with this technology. Obtaining these images requires specific weather conditions. Time and expertise are needed in order to interpret the data. Images captured by sensors with different sensitivity levels provide massive amount of information in the form of an image cube. Because each pixel covers a specific area, which may be composed of different materials, some pixel spectra are comprised of a mixture of spectral signatures of those materials. In addition to the large amount of data, noise will also affect the results. These factors make it difficult to analyze the data manually. The noiseless signature of a material is called the endmember of that material [74]. Several techniques are regularly used to interpret the raw data and predict the existing materials or estimate the distribution of different minerals on a hyperspectral image. These techniques work quite well on simple data. In more com-

plicated cases with numerous different materials, however, it is not possible to recognize the identity of each single pixel of the image. There are many applications for remote sensing. Exploring different rocks and minerals within an area are some of these remote sensing applications [72]. In many cases, examining constitutive materials of a region of earth is too difficult for logistic reasons, and therefore remote sensing can contribute in identifying the minerals, rocks and other materials of that area.

Mineral identification problems can be stated as a classification problem or a clustering one. In this research study, Support Vector Machines (SVM) and Self-Organizing Maps (SOM) are applied as classification and clustering techniques. SVM is one of the most popular classification methods used for hyperspectral classification [57] and it has been used in different variations in this field of study [12, 66]. Many potential improvements can be applied to increase its performance. Improving this powerful classification method for hyperspectral data by paring it with Principal Component Analysis (PCA) and applying Error-correcting output codes (ECOC) on it, is the motivation for this research study.

SOM is not a widely used technique in hyperspectral mineral identification. There are statistical and geometrical approaches due to unmixing and clustering hyperspectral data [7]. In this study, SOM is selected as the unsupervised learning method [42] to assess the advantage of supervised learning versus unsupervised learning. When SVM is applied to real data as a classification method, it is important to use the least possible number of samples for training. Extracting the most useful aspects of the spectra is another important matter. Principal Component Analysis (PCA) is applied due to extract useful spectral features. One of the goals of this thesis is to extract the most informative features and increase the accuracy of classification on small training sizes. Cuprite data which is a well-defined hyperspectral dataset [61] is used as benchmark. A hyperspectral dataset from Baffin island represents a less ideal dataset used for evaluating the proposed method.

1.1 Goals

The main goal of this research study is to evaluate a classification method against a clustering technique. SVM is a common classifier that has been used in this field [28, 24, 2, 81]. We try to improve the common SVM structure by applying ECOC. Self-organizing maps (SOM) are frequently introduced as a fairly strong clustering method in situations in which the number of clusters is not known in advance [42]. Three approaches contribute towards the main goals of this research:

1. Improving support vector machines implementation.
2. Applying self-organizing maps.
3. Comparing the results of the classifier and the clustering method with the improved classification approach.

1.2 Thesis Structure

The structure of this thesis is as follows. Chapter 2 explains background information on hyperspectral remote sensing. This chapter also reviews literature for different feature extraction and classification methods that have been used in the area of hyperspectral classification. Chapter 3 describes the implemented methodology in detail. Chapter 4 explains the data and all of the experiments and analysis the results from different runs and finally, Chapter 5 presents the conclusions and describes future research opportunities.

Chapter 2

Background and Literature Review

In this chapter, background information on the aspects of earth science and computer science used throughout this research is reviewed. The earth science terminology is needed to describe the problem this work aims to solve. The machine learning tools will also be discussed in the following chapters.

2.1 Remote Sensing

Remote sensing is a field of study which acquires information from an object without physical contact with it [74]. This is done by sensing the emitted electromagnetic energy, and processing and analyzing the data [38]. The field of remote sensing is broad and encompasses vision, astronomy, observing Earth from a distance, sonar and even medical imaging. Remote sensing can be addressed as an instrument-based technology used to acquire and measure spatially organized data on electromagnetic energy reflected by objects on the Earth's surface. In this thesis, the discussions and applications of remote sensing will be restricted to applications within earth sciences. Hyperspectral imaging is a type of spectral imaging which has been used broadly in remote sensing. It collects

Figure 2.1: Graphic representation of hyperspectral data [21].

and demonstrates information from across the electromagnetic spectrum [13]. Human eyes see visible light in wavelengths between 380 nm and 750 nm. This bandwidth is pretty narrow compared to some hyperspectral imaging instruments which record information in the wavelength range of 380 nm to 2500 nm [70]. This range is divided into narrower ranges, each with a restricted wavelength, referred to as spectral band. The three-dimensional representation of these images constructs a hyperspectral image cube, which is used for processing and analysis. A two-dimensional representation of a hyperspectral cube is shown in Figure 2.1. One of the common airborne sensors is NASAs Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) which is able to acquire more than 200 bandwidths for a single location [30]. Figure 2.2 shows the schematic of hyperspectral imaging. For a specific object, each of the spectrometers records the reflectance in a narrow range. Putting these narrow wavelength slices together reconstruct the entire spectra of the target. Each material exhibits unique absorption characteristics. Hence, its spectrum could be considered as a signature for it. In earth sciences, the reflection

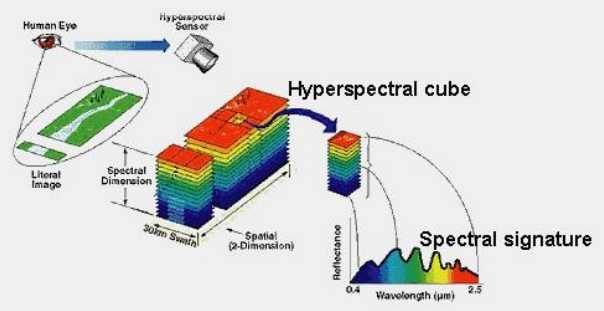


Figure 2.2: Hyperspectral imaging schema [27].

characteristics of each mineral measured with precise instruments in laboratory is called endmember of that mineral [74].

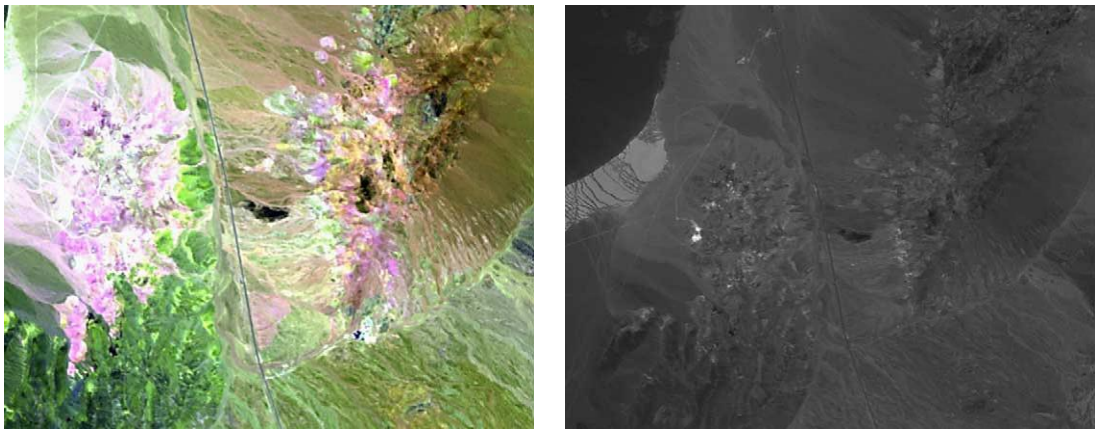
Cuprite is a standard hyperspectral case study in earth sciences. Cuprite is an area located in Nevada, US and its information have been captured by AVIRIS. Figure 2.3 shows a false colour RGB image and the reflectance image at 2229 nm bandwidth. AVIRIS and other hyperspectral imaging devices consist of hundreds of spectrometers where each of them is sensitive to a specific reflectance wavelength. Hyperspectral imaging produces a large amount of information and it has its advantages and disadvantages [5]. The data provided by hyperspectral imaging helps to identify constituent materials of an area. On the other hand, finding relevant and helpful information is difficult due to an issue known as the "curse of dimensionality" [4]. This refers to issues that arise in analyzing data in high-dimensional spaces that do not occur in low-dimensional data. Increasing the dimensionality of data increases the volume of the space, so the available data becomes sparse [3]. Another problem is the effect of noise. There are two effects, which add noise to the radiance spectrum: sensor offset and atmospheric scattering. As the sensors are not perfectly precise, sensor offset, which is the internal instrument noise adds some levels of error to the procedure of acquiring images. Particles suspended in the atmosphere cause redirection of energy which is called atmospheric scattering. The wavelength of incoming radiation and the depth of atmosphere that the radiation must travel through, the

number of particles in the atmosphere and the size of the particles affect the atmospheric scattering.

One way of extracting information is via machine learning with the cooperation of the scientists in both fields (computer science and earth sciences). Biederman's work [6] could be mentioned as one of the first examples of this collaboration. Removing the noise and finding the important and useful features is the main issues that computer science contributes to hyperspectral remote sensing [62, 34].

2.2 Pattern Recognition Tools

Different pattern recognition techniques and tools are used in the experiments discussed here. These techniques and the relevant concepts are explained in this section and will be addressed in the methodology section.



(a) False-colour RGB image. (b) AVIRIS image, 2229 nm.

Figure 2.3: Cuprite, Nevada and its AVIRIS image 2229 nm.

2.2.1 Feature Extraction

In most of real pattern recognition applications, the data needs to be transformed into a new space of variables to make problem-solving easier [8]. The purpose of extracting features is to retrieve the features which best represent the objects. Feature extraction naturally leads to dimensionality reduction [75]. There are some problems which make feature selection essential as a part of preprocessing. Large numbers of features, redundant features and less informative features are problems, which will reduce the accuracy of later processes. Feature extraction encompasses a wide range of algorithms and techniques. Feature extraction techniques are mostly defined as transformations. The extracted feature can be a linear combination of raw features or a non-linear combination in more complicated feature extractors.

2.2.1.1 Principal Component Analysis

Principal component analysis (PCA) is a widely used feature extraction method, which has found application in many fields such as face recognition [40] and hyperspectral image compression [22], and is a common technique for identifying patterns in data of high dimension [8]. PCA represents the data in a specific way to distinguish their similarities and differences. Since graphical representation for data with more than three dimensions is not available and patterns in high-dimension data can be hard to find, PCA is a powerful tool to make the data understandable and easier to analyze. The main advantage of PCA is that it compresses the data and avoids redundant features with minimum loss [17]. Following five step represents how PCA can be applied to a set of data:

- **Step 1: Subtracting the mean:** This produces data with mean of zero in all dimensions.
- **Step 2: Calculating the covariance matrix:** Each element of this matrix shows the correlation of two dimensions of the dataset.

- **Step 3: Calculating the eigenvectors and eigenvalues of the covariance matrix:** Since the covariance matrix is square, calculating the eigenvectors for this matrix is possible.

- **Step 4: Choosing components and forming a feature vector:** The feature dimensions with larger eigenvalues contain more information and are considered as principal components. Dimensions with smaller eigenvalues carry less information. The eigenvectors need to be ordered by eigenvalues. By choosing a subset of eigenvectors, the data is compressed and will be reduced. In this way the dimensions which are not carrying much information will be ignored.

- **Step 5: Transferring data to the new space:** By multiplying the data into the matrix of selected eigenvectors the data will be transformed to the new space.

2.2.2 Support Vector Machines (SVM)

In the supervised classification process a portion of data samples is used to model the data. This set of data is called a training set, and it is given to the learning algorithm to create a data model. The two-class classification problem is the easiest classification problem. In these cases classification algorithms should find a line or a curve (in one or two dimensional data) or a hyperplane to separates the class samples. Support vector machines (SVM) [10] were introduced to the statistical learning domain [78] for regression and classification two decades ago. They have also been used for classification of hyperspectral data [31, 56, 49]. This technique finds an optimal separation surface between the class samples. Generally there are lots of surfaces, which can separate the classes. Despite of similar classification methods (multilayer perceptrons) SVM tries to find the *optimal* hyperplane with the most generalization ability. SVM finds the hyperplane, which has the most distance with support vectors. Support vectors are the training samples which are nearest to the other class's samples. They are located at the border of their class. The

mathematical formulation of SVM is available in [10, 77].

2.2.2.1 Basic SVM

Basic SVM, as proposed by Vapnik in [78], provides a solution for a linearly separable two-class problem. The SVM defines a hyperline, which has the maximum generalization ability. This classifier has the widest margin. The margin is defined as the distance of the hyperplane to the nearest support vector. Basic SVM cannot be applied for non-linearly separable or multiclass problems. Figure 2.4 shows a representation of a maximum margin separating hyperplane.

Figure 2.4 shows a set of n data samples of two classes. In this sample two classes are linearly separable. The procedure of finding the maximum margin classifier (by support vector machine) for this data isasn follows: Any hyperplane could be described as a set of points x satisfying $w.x - b = 0$,. The parameter $\frac{b}{\|w\|}$ is the offset of the hyperplane. Considering the data is linearly separable, there are two hyperplanes that separate the

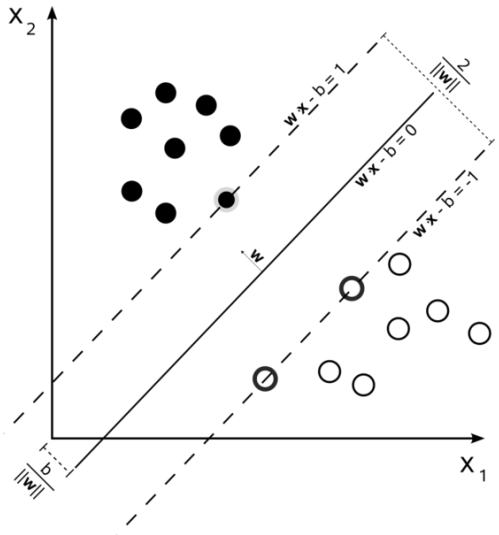


Figure 2.4: Support vector machine schema [16].

data, and there are no points between them. SVM tries to maximize their distance. The area bounded by these hyperplanes is called "the margin". These two hyperplanes can be written by the equations $w.x - b = 1$ and $w.x - b = -1$. The distance between these two hyperplanes is $\frac{2}{\|w\|}$. As the data points are not supposed to fall into the margin, the following constraint should be a added:

$$\begin{cases} w.x_i - b \geq 1 & \text{for } x_i \text{ of the first class} \\ w.x_i - b \leq -1 & \text{for } x_i \text{ of the seond class} \end{cases}$$

or it can be restated as $y_i(w.x_i - b) \geq 1, \quad$ for all $1 \leq i \leq n$. Therefore the problem of finding the hyperplane $w.x - b = 0$ with maximum margin is reduced to minimize $\|(w, b)\|$. In this optimization problem $\|w\|$ could be replaced by $\frac{1}{2}\|w\|^2$. This is a quadratic programming optimization problem: $\underset{(w,b)}{min}\frac{1}{2}\|w\|^2$ subject to the constraint (for any $i = 1, \cdots, n$) $y_i(w.x_i - b) \geq 1$. Lagrange multiplier is a strategy to find the minimum or maximum of a function with equality constraints [80]. By introducing Lagrange multipliers α, this constrained problem can be expressed as

$$\min_{(w,b)} \max_{\alpha \geq 0} \{\frac{1}{2}\|w\|^2 - \textstyle\sum_{i=1}^{n} \alpha_i [y_i(w.x_i - b) - 1]\}$$

At this point one saddle point is the required answer. This is a standard quadratic programming problem and according to the Karush-Kuhn-Tucker's conditions [44] the solution is a linear combination of data samples.

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i \tag{2.1}$$

Only a few α_i will be greater than zero. The corresponding x_i are the support vectors, which lie on the margin and satisfy $y_i(w.x_i - b) = 1$. as the $y_i = \pm 1$ and nonzero, therefore

$$w.x_i - b = 1/y_i \Longleftrightarrow b = w.x_i - y_i$$

And (w, b) indicates the separator hyperplane with maximum margin.

2.2.2.2 Non-linear separability

In non-linearly separable data, there is not any line which can separate classes. To enable SVM to classify non-linearly separable data, two general approaches are used: soft margin SVM and kernel SVM.

In soft margin SVMs there is a limited tolerance of error in order to converge to a separation hyperplane [82]. In this approach, the classifier is not a curve but by accepting some samples located in the margin makes it possible to attain a separation hyperline. The generalization ability of this classifier is less than that for basic SVMs.

Another technique to handle non-linearly separable classification problems is to use kernel SVMs. Kernel transfers data from input set S into an inner product space V. The data in new space has more dimensions compared to the original space. A suitable kernel makes the data linearly separable in the new space [9, 60]. Effect of a general kernel K on vectors x_i and x_j from set S can be expressed as

$$K(x_i, x_j) = \langle \varphi(x_i), \varphi(x_j) \rangle \tag{2.2}$$

Using a kernel produces new dimensions for the data. This transformation to a higher dimension space increases the chance of finding a dimension which the data is linearly separable on it. Figure 2.5 shows a general representation of applying a kernel on a non-linearly separable data. In this figure, the left image shows the data in the original space and the right image shows two dimensions of data in the new feature space.

The polynomial (homogeneous) kernel could be mentioned as the simplest kernel for SVM. This kernel is defined as

$$K(x_i, x_j) = (x_i.x_j)^d \tag{2.3}$$

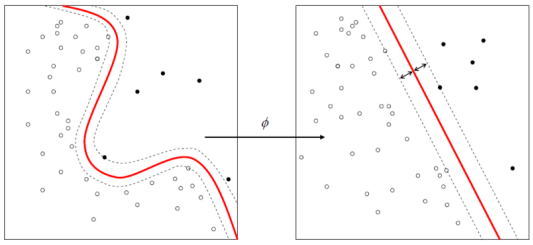


Figure 2.5: Mapping non-linearly data using kernel function [1].

2.2.2.3 Multi-class SVM

There are numerous binary classification methods like decision trees [73], Bayesian networks [37] and SVMs which are successful in binary classification. But SVM classification ability is not limited to two-class problems. There are algorithms to extend binary classifiers to multiclass classification. The one-vs-all classification [68] and one-vs-one classification [35, 33] are two popular and simple algorithms for this purpose.

One-vs-all strategy consists of constructing a SVM for each class. These SVMs are trained to distinguish a class from all other classes. One-vs-all converts a $C-$class problem to C binary classification problems. According to one-vs-all strategy, SVM_i which separates class i from other classes considers class label of 1 for data samples belong to class i and 0 for all other samples.

One-vs-one strategy converts $C-$class problem to $C(C-1)/2$ binary classification problems. In this method there is one SVM to distinguish each class from another. According to this strategy there is one SVM_{ij} to separate class i from class j. The other class samples do not considered in training this SVM. This approach is more accurate than one-vs-all because it separates each two classes specifically. Although the number of SVMs getting trained is much more in one-vs-one strategy, but each of these SVMs should solve easier problems. Therefore, the runtime of one-vs-one strategy is less than one-vs-all [59].

Error-correcting output codes (ECOC) [20] is the multiclass strategy that has been

used in this research study. ECOC is more complicated than the other strategies, but it is more efficient. Similar to other multiclass strategies, ECOC generates a bit vector for each class. This process is best explained using an example. Table 2.1 contains an example of an ECOC code generated for a four-class problem. In order to classify a four-class problem with a binary classifier (like SVM) using ECOC, seven classifiers are needed. Table 2.1 shows the regular labeling and ECOC class labeling for each class. In this table, the bit vector assigned to each class label shows what each classifier is given as class label for the set of data. For example, in the first row, the first class is introduced to the first three classifiers as class one and has been introduced to $4th$ and $7th$ classifier as class 0. In this way each of these seven binary classifiers should separate $1s$ from $0s$.

The most important advantage of this coding system over the two other strategies is error correction. Each class bit vector is different by other class bit vectors at least in three bits, and it helps to identify the class label of a data sample, even if one of the classifiers made a wrong decision about it. An example demonstrates the benefit of this feature. Consider a data sample that belongs to the first class. If the first classifier misclassifies it, the generated code will be different by the first class's bit vector in one bit. However, it will still be different to the other three bit vectors in two bits, and thus there is only one closest code to it. Therefore ECOC enables the decision system to make a right decision even though one classifier misclassified it.

Table 2.1: Binary codes generated by ECOC for a four-class problem.

Regular labels	**ECOC labels**						
1	1	1	1	0	1	1	0
2	1	0	0	1	0	1	0
3	0	0	1	1	0	1	1
4	1	1	0	1	1	0	1

2.2.3 Self-organizing Maps

Unsupervised learning refers to finding a hidden structure in unlabeled data. In unsupervised learning, which is also called clustering, learning processes consists of input data without corresponding target values [8]. Clustering categorizes the data into groups based on the similarities and differences of features.

The self-organizing map (SOM) [41], which is also called a Kohonen map, is an unsupervised neural network learns based on winner-take-all learning strategy. In this competitive strategy, each neuron inhibits others in order to stay active [52]. Producing a two dimensional visualization of data is one of SOM's advantages. SOM's most important feature is that it categorizes the data without knowing the actual number of categories (clusters) [42].

A SOM consists of a number of nodes (neurons). In addition to having specific coordinates, each node has some weights. There are as many weights as dimensions of input for each node. The weights are initiated randomly. During the training process, each data sample chooses the best matching unit (BMU). BMU is the node which its weights has the minimum distance with the input vector. Euclidean distance is usually used for calculating the distance:

$$dist(x, w) = \sum_{i=1}^{n}(w_i - x_i)$$

where x is the $n-$dimensional input vector and w is the given node's weight vector. The weight on the BMU and its neighbors are adjusted toward the input vector. This adjustment is done using a learning rate. The learning rate and neighborhood size are decreased during the training process. Adjusting process of weights can be expressed as:

$$w_i(t+1) = learningrate(t) \times (w_i(t) - x_i)$$

where $w_i(t)$ is the i^{th} weight of node at time t.

The pseudo code for the SOM is as follows:

Input : InputVectors, $iterations_{max}$, $learnRate_0$, $neighborhoodSize_0$, $lattice_{width}$, $lattice_{height}$
Output: WeightVectors
WeightVectors ← InitializeWeightVectors($lattice_{width}$, $lattice_{height}$, InputVectors)
For($i = 1$ to $iterations_{max}$)
  $learnRate_i$ ← CalculatingLearningRate(i, $learnRate_0$)
  $neighborhoodSize_i$ ← CalculateNeighborhoodSize(i, $neighborhoodSize_0$)
  $Vector_i$ ← SelectInputVector(InputVectors)
  Bmu_i ← SelectBestMatchingUnit($Vector_i$, WeightVectors)
  Neighborhood ← Bmu_i
  Neighborhood ← SelectNeighbors(Bmu_i, WeightVectors, $neighborhoodSize_i$
  For($Weight(i) \in$ Neighborhood)
    For($Weight(i)_{attribute} \in Vector_i$)
    $Weight(i)_{attribute} \leftarrow Weight(i)_{attribute} + learnRate_i * ((Vector(i)_{attribute}$ -
    ($Weight(i)_{attribute}$)
    end
  end
end
Return(WeightVectors)

2.3 Literature Review

During the last two decades several machine learning approaches have been applied to classify hyperspectral images [13]. These algorithms can be divided into two main categories: (i) feature extraction algorithms and (ii) classification algorithms. Feature extraction algorithms try to deal with the curse of dimensionality corresponding to large numbers of noisy features. Classification algorithms have been applied to identify the contents of all pixels or tracing some particular material in hyperspectral data [53].

2.3.1 Feature Extraction Methods

Different versions of PCA are popular feature extraction methods for high-dimensional data [39]. The coordinates of the eigenvector elements are called principal

components. The directions of the first n eigenvectors corresponding to the biggest n eigenvalues cover as much variance as possible by n orthogonal directions. Since the transformed features are sorted by the variance of data in them, in many applications the first features contain the most interesting information. For instance, in data compression, data is projected onto the directions with largest variance to retain as much information as possible. Moreover, in de-noising small variances are deliberately discarded [58, 19]. In addition to the basic PCA, there are new generations of PCA which are designed to improve the performance of feature extraction in more complicated cases. As outlined below, these have been broadly applied to hyperspectral datasets.

2.3.1.1 Noise-adjusted Principal Component Analysis

One type of PCA is noise-adjusted principal components (NAPC). NAPC tries to handle noisy data. In order to do so the signal-to-noise ratio (SNR) is taken into account as well as the variance. The principal components resulting from a maximum variance-based PCA do not necessarily represent image quality. Applying maximum noise fraction (MNF) will cause the PCA to sort features by signal-to-noise ratio rather than variance [29]. The MNF was presented later in [46] as a noise-whitening processing. One of the major disadvantages of this approach is that the noise covariance matrix must be estimated accurately from a priori knowledge, which is not possible in most cases. Another issue is that the factor of interference is not taken into account in MNF or NAPC in which the interfering effect tends to be more serious than the noise in hyperspectral images [14, 48].

2.3.1.2 Kernel PCA

The kernel PCA, as a nonlinear feature extractor, has been proven to be a powerful preprocessing tool for classification algorithms. It can also be considered as a natural generalization of linear principal component analysis. Kernel PCA first maps the data onto a new domain using the kernel. By choosing a suitable kernel, extracting features in

new space would be easier. As PCA basically is a linear transformation, using a nonlinear transformation as a kernel would be significantly helpful in hyperspectral data classification [54]. Applying a kernel to transfer features to a new space is essential for hyperspectral data and other high-dimensional data. Kernel PCA is a useful method to extract features for nonlinearly separable data. However, using kernel classifiers like SVM is another alternative for transforming data.

2.3.2 Classification Methods

2.3.2.1 Artificial Neural Networks

Artificial neural networks [69, 47] are modeling systems inspired by the brain's structure. They learn patterns from experiences and predict new cases. The most popular artificial network is the multi-layer perceptron (MLP) [71]. MLP is a supervised learning tool. The simplest structure for an MLP consists of one input layer, one output layer and at least a hidden layer of neurons in the middle. The number of neurons in the input layer is the same as number of features and all of them are connected to the neurons in hidden layer with coefficients called weights. In fact these weights regulate the intensity effect of each feature in the decision process. Artificial neural networks have always been popular classifiers [11, 51]. They have been successful in modeling the distribution of land cover or vegetation based data in remote sensing [64, 25]. They have been applied in other remote sensing studies. Species richness [32], biomass, productivity [45] or conservation priority of habitats [23] are some examples of the wide application of MLP in remote sensing.

One important advantage of MLP is their non-parametric nature. This feature made them able to deal with almost all kinds of data without considering its statistical properties. On the other hand, the results of neural networks are sometimes costly to interpret. The output of the network cannot represent the data clearly. Another disadvantage of artificial neural networks is the high risk of over-fitting. Over-learning (over-fitting) means

that the classifier becomes adapted to the training samples. Therefore it loses the generalization ability. This usually happens in large networks with complex functions [76].

2.3.2.2 Genetic Programming

Genetic programming (GP) is an evolutionary algorithm inspired from biological system. GP not only is an optimization tool but also it could be used as a classifier. GP is a specialization of genetic algorithm (GA) [26] proposed by Koza [43]. One successful approach is training a single GP for each class [70]. In this experiment it is applied to Cuprite data, there is one GP system for each class. It means that each GP system considers samples of one class as positive samples and the other samples as negative ones. One specific feature of GP which makes it suitable for hyperspectral classification, is its ability in evaluating the usability of features and increasing or decreasing the effect of each feature in classification process.

2.3.2.3 Support Vector Machines

A Support Vector Machine (SVM) proposed by Vapnik in 1979 [79] is a classification technique. Similar to artificial neural networks, SVMs are non-parametric learning technique, and so there is no need for priori knowledge of data distribution. SVMs have recently been widely used for the classification of hyperspectral images [63], which SVM is used as the major classification tool in remote sensing applications. Assigning weights to the misclassified samples and minimizing the total error was one practical way to apply SVMs on hyperspectral data [49]. SVM appears to be a robust alternative for pattern recognition with hyperspectral data. Since the method is based on a geometric point of view, no statistical estimation is necessary. SVM outperforms the classical supervised classification algorithms, such as maximum likelihood, when the number of spectral bands increase or the number of training samples is limited. Among different common pattern recognition methods such as maximum likelihood, K-NN and neural

network (RBF and MLP), SVMs have shown the best performance in solving the remote-sensing problems [36, 56, 57].

Chapter 3

Methodology

The implemented system consists of several steps, including receiving raw data, preprocessing, machine learning processes, and post-processing stages. This chapter provides a detailed explanation of these steps.

3.1 Data Preprocessing

The following proposed method is applied to hyperspectral data. These data were provided in the format of multiple 8 bit (0-255) images, one for each spectral band, with the same X/Y dimension. The measured reflection intensity within each band is scaled to a value between 0 and 255 in each image. This way the gray level intensity of each pixel in each image shows the measured feature of the corresponding area. For one of the data sets (Cuprite) there is one RGB image containing information about the class, which corresponds to each pixel. This image is referred to as the ground truth image. In the first step, the data is extracted from raw images. Each data sample consists of the pixel intensity values of every pixel located in the same relative position in all images. Reading all the images and saving the information as a set of feature vectors, creates a dataset.

In this dataset all data samples have the same number of features (corresponding to the number of images).

In the first step, data is normalized. Since the minimum and maximum possible values for each feature is 255, by dividing all values by 255 the data will be normalized. The minimum and maximum values of some feature dimensions may be different and dividing them by 255 may change the variance of the data in those feature spaces. Nevertheless, this makes it possible to retrieve the original data in case if necessary. It should be mentioned that only pixels or samples for which a classification is known are used in the experiments because only these pixels can be used to evaluate the accuracy of the methods. This means that pixels with a zero value (black) are not included.

3.2 Feature Extraction Using PCA

The hyperspectral data typically have more than one hundred features (spectral bands). Such a large number of features will cause some restrictions for machine learning algorithms. Some classifiers (even strong ones like SVM) are very sensitive to the number of features. In order to reduce the number of features and also identify the informative ones, PCA is used on these data. Matlab's 2011 [55] "statistics Toolbox" is employed for the implementation of PCA algorithm. Specifically *princpmp* function of Matlab performed the PCA transformation for this experiment. By applying PCA, in addition to transformation of data, eigenvalues will also be calculated. Eigenvalues act like indexes to measure the usability and importance of each feature dimension. The data in the first dimensions of PCA is more distributed than the features in dimensions with smaller eigenvalues. The classification method will be applied on PCA-transformed data. SVM is the selected classifier for this purpose. SVMs are highly sensitive to the number of features and failure in finding separator line in even one feature space prevents it from converging to a hyperplane. Therefore applying such a feature extractor is essential. In order to find the best set of features, various numbers of them were considered in different

experiments (for classification and clustering). A comprehensive set of features carries informative features and also avoid redundant information.

3.3 Classification Using SVM

Classifying a multiclass dataset using SVM has several steps. It could be stated as a 5 step process:

1. Assigning binary class labels: In order to classify multiclass data, a set of SVMs should be trained. Data samples need to have binary class labels. The ECOC has been used to generate the binary class labels for each class according to the generated labels. It is also responsible for regrouping the class samples to prepare a two-class classification problem for each SVM.

ECOC algorithm assigns a bit vector to each class label. In each bit vector, the ith bit shows the new class label of each class for ith classifier. The classes with label 1 are considered as one group and the ones with label 0 are considered as another group. The number of needed SVMs depends on the length of bit vectors generated by ECOC. The ECOC algorithm was implemented in Matlab 2011.

2. Training and test sets: For each experiment there is a training set and a test set. The portion of data that is the training set is excluded from the eventual testing. All samples not part of the training set are considered as the test set. Various sizes of training sets have been tried, which is explained later in Chapter 4.

3. Finding kernel: Considering the fact that the data is not linearly separable, finding an appropriate kernel is an important task. Even using soft margin SVMs, the classifier tolerates certain amount of errors. If an inappropriate kernel has been chosen, the SVM fails to converge to any separator. Among the kernels provided for SVM in Matlab Bioinformatics Toolbox (linear, quadratic, polynomial, MLP and RBF) RBF is the only kernel

which showed acceptable results.

Gaussian radial basis function kernel (RBF) is the most popular kernel for SVM [15]. RBF kernel simply shows the similarity of two vectors. RBF which is also known as squared Euclidean distance between two feature vectors is defined as:

$$K(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2}) \tag{3.1}$$

where σ is a free parameter.

4. Optimization solution algorithm: As explained in chapter 2, finding a hyperplane could be defined as a optimization problem. There are common algorithms such as quadratic programming (QP), least squares error (LS) and sequential minimal optimization (SMO) for solving optimization problems.

QP is a very time consuming (about 25 times more than SMO) and in numerous cases the program doesn't converge to a hyperplane in an acceptable time. It also expensive in case of computer hardware especially RAM. LS achieves to a hyperplane but is is not successful (as LS does not support soft margin).

SMO is the selected algorithms to find the best separating hyperplane. It is much faster than QP and doesn't need much computer memory. It also supports soft margin. According to SMO, the quadratic programming problem of training a SVM could be expressed as

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j K(x_i, x_j) \alpha_i \alpha_j, \tag{3.2}$$

subject to $0 \geq \alpha_i \geq C$, for $i = 1, 2, \cdots, n$, and $\sum_{i=1}^{n} y_i \alpha_i = 0$.
which C is a SVM parameter and $K(x_i, x_j)$ is the kernel function and α_i are the Lagrange multipliers.

After selecting suitable kernel and optimization algorithm, SVMs get trained on the training sets.

5. Evaluating classifier: After training the SVM set and getting the results of the test set, the accuracy of the classifier on each class is measured. For each testing sample, each SVM returns a bit as class estimation. The results of all SVMs create a bit vector for each sample. The answer vector and the target vector (assigned by ECOC) are compared. If they are the same or different in one bit, those samples will be considered as correct answers. ECOC guarantees that the target bit vectors are different at least in three bits. Therefore, one misclassification could be corrected. This feature helps to increase the overall accuracy of the classification result.

The overall performance of the classification system is the average accuracy of classification on all classes regardless of the number of samples belonging to each class. The accuracy of classification of each class is calculated by dividing the number of correctly classified testing samples per each class by the size of testing set of that class.

The classification process is applied with different configurations on each data. These different configurations consist of different sizes of training sets and different number of features. In order to make sure that the results are reliable, 30 runs have been carried out for each configuration set. In each run, the training set have been selected randomly from data samples. *svmtrain* and *svmclassify* functions from Matlab 2011 "Bioinformatics Toolbox" performed SVM classification. Table 3.1 summarized the configuration parameters used for SVM.

3.4 Clustering Using SOM

In the clustering process, SOM will be applied to PCA transformed and to non-transformed data. The SOM Algorithm was implemented in Matlab 2011. The SOM algorithm is simple and does not contain many variables. The number of initial nodes and the structure can be changed. SOMs with different numbers of initial nodes and different

Table 3.1: SVM parameter values.

Parameter	Value
Maximum tries	100,000
Kernel function	RBF
Optimization solution algorithm	SMO
RBF sigma	1

numbers of features were used in order to evaluate their ability in categorizing data without the knowledge of the number of classes. The number of iterations selected was 200. Best matching unit (BMU) refers to the node selected as the best matching for each data sample. In order to find BMU, Euclidean distance function is applied. Euclidean distance function is given as:

$$Dist = \sqrt{\sum_{i=0}^{i=n}(w_i^2 - x_i)}$$

where x is the current input vector and w is the node's weight. The radius of a node's neighborhood changes throughout the iterations. The neighborhood radius shrinks over the time. The neighborhood at iteration t is noted by $\sigma(t)$ and changes as

$$\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\lambda}\right)$$

where σ_0 denoted as the initial width of lattice and $\lambda = (200/\log(\sigma))$. The learning rate at each iteration is denoted by $leariningRate(t)$ and the decay of learning calculated as

$$learningRate(t) = learningRate_0 \times \exp\left(-\frac{t}{\lambda}\right)$$

and $learningRate_0$ is set as 0.1 in this experiment.

Table 3.2 summarizes the values of parameters which have been used in SOM. The values of maximum iterations and initial learning rate are chosen based on initial results

Table 3.2: SOM parameter values.

Parameter	Value
Maximum Iterations	200
Distance function	Euclidean
Initial learning rate	0.1
Initial neighborhood size	3

of SOM. For distance function and initial neighbourhood size, the default values are used [18].

After finishing the SOM's training phase, the accuracy of the categorization will be evaluated. As the actual class label of each data sample is available it is sufficient to find a category that estimate each class the best. This is done by a maximal matching process between categories obtained by SOM and actual classes. In order to do this, one category should be assigned to each class. The method which is employed in this experiment is as follows. A class will be assigned to a category if that category is the best representative of the class in question. In other words, the category should contain more of the class's samples than all other categories making that category the best representative of that class. This process is repeated for all classes.

Figure 3.1 represents different stages of the methodology in this research study.

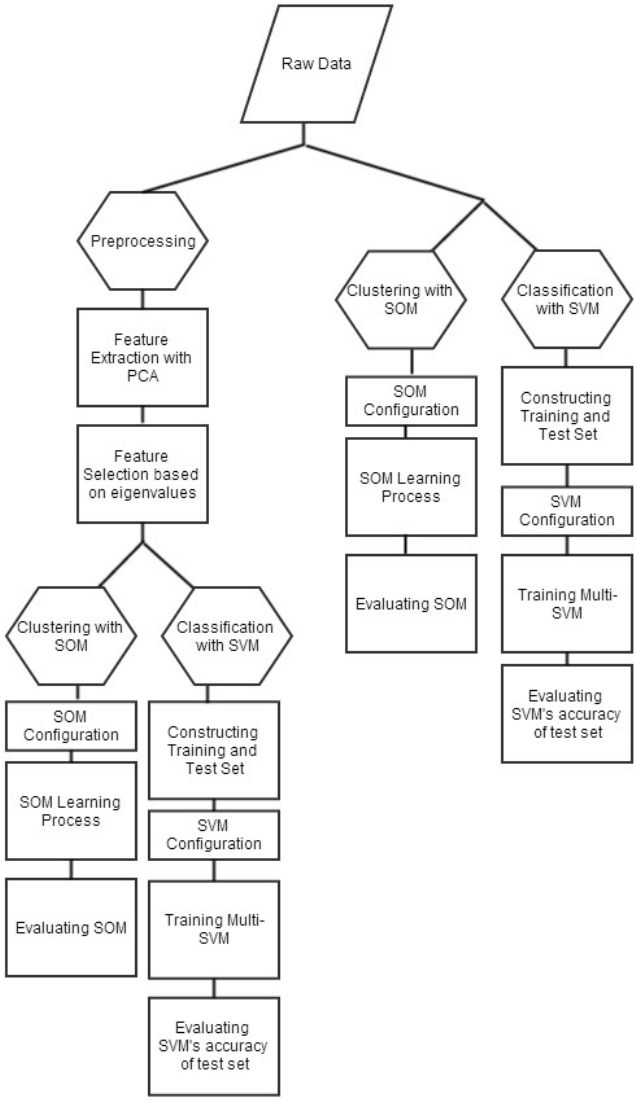


Figure 3.1: Methodology flowchart

Chapter 4

Datasets, Experimental Results and Discussions

This chapter is divided into two sections. In the first section, the hyperspectral datasets which have been used are discussed. The second section presents and discusses the experimental results, starting with a summary of experiments carried out.

4.1 Hyperspectral Datasets

The classification and clustering algorithms introduced in the methodology chapter have been applied to two datasets for evaluation. The Cuprite data is a well-studied benchmark [67] to test hyperspectral classification tools and the Baffin Island data represents a realistic Canadian earth science problem.

4.1.1 Cuprite Dataset

Cuprite Data is derived from the AVIRIS hyperspectral data set by Neville at al. [61] in 1998. The original data was obtained from Cuprite, Nevada on June 12, 1996. It includes

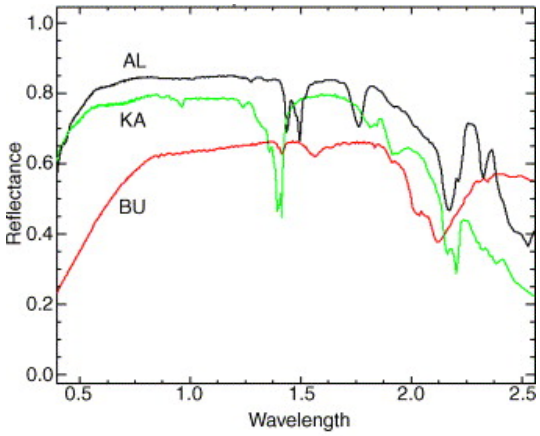
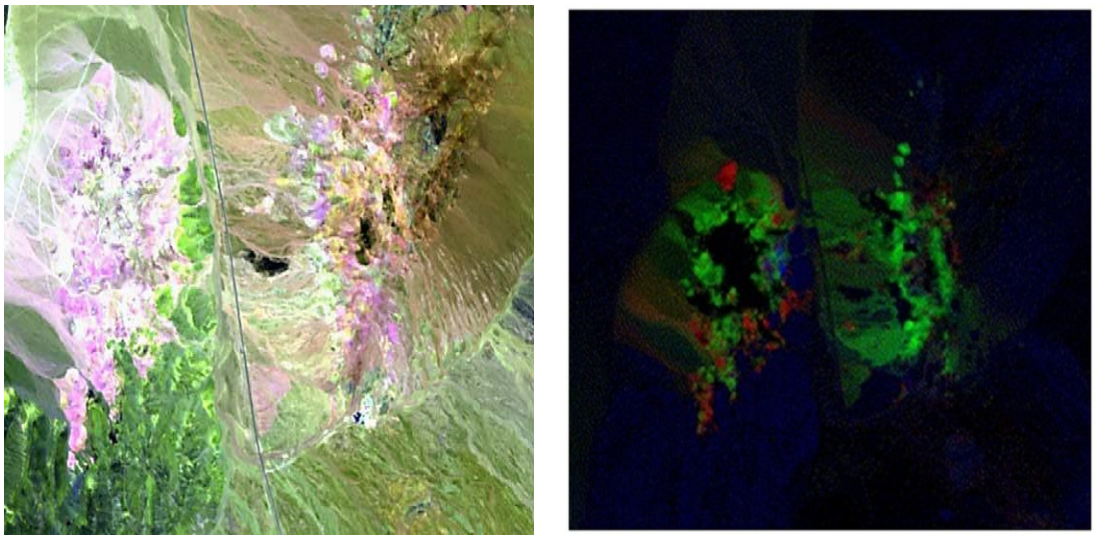


Figure 4.1: Spectra for alunite (AL), kaolinite (KA), and buddingtonite (BU) [70].

wavelength regions from 0.38 to 2.50 microns with the interval of 16.2 microns. The mineral fraction map derived from this data was analysed by Nevile [61]. In this dataset three clay minerals, alunite (AL), kaolinite (KA) and buddingtonite (BU) are identified. Figure 4.1 shows the laboratory spectra of these three minerals. Ross et al [70] selected 168 bands in the range of 0.48 microns to 2.438 for their experiments. These 168 bands of data are used in this experiment. Figure 4.2 shows a false-colour composite image of the area (left) and the mineral fraction map also known as the ground truth (right). The mineral fraction map is also the target image for the experiments applied in this study. It is an RGB image in which the value of a pixel (red, green and blue) demonstrates the intensity of alunite (AL), kaolinite (KA) and buddingtonite (BU) respectively in that area. Having a channel dedicated to each class made it an appropriate benchmark for classification methods. Most of the pixels in Figure 4.2 are impure and composed of a mixture of several minerals. Because each mineral is assigned to a separate channel, it is possible to choose pixels with specific levels of purity. The difference of the intensity of the major mineral from the other minerals in each pixel is considered the purity index.

In order to take advantage of this, four separate datasets were extracted from Cuprite data with different thresholds based on the purity index. Each dataset includes the pixels with purity indices higher than the threshold set for that dataset. This creates datasets with different classification difficulties. The purity index has been used to test the ability of methods to deal with different classification difficulties. Four thresholds, arbitrarily chosen were used in this experiment. The concept of setting thresholds comes from the similar work of Ross [70]. In that experiment he considered samples with purity indexed more than a certain amount as positive samples for samples' major class. The major mineral of p_i is signified as $p_{i,max}$

$$pixel_i = (p_{i,1}, p_{i,2}, p_{i,3}) \tag{4.1}$$



(a) False-colour RGB image. (b) Mineral distribution ground truth.

Figure 4.2: Cuprite, Nevada and its mineral distribution ground truth.

Table 4.1: Cuprite derived datasets' information.

Name	Threshold	Total samples	Buddingtonite samples	Kaolinite samples	Alunite samples
Cuprite Dataset1	256	3065	360	2636	69
Cuprite Dataset2	192	7551	972	6452	127
Cuprite Dataset3	128	15860	2007	13435	418
Cuprite Dataset4	64	81482	5346	28206	47923

and

$$p_{i,max} = max(p_{i,1}, p_{i,2}, p_{i,3}) \tag{4.2}$$

The $dataset_d$ will contain the samples if:

$$\sum_{j=1}^{3}(p_{i,max} - p_{i,j}) > threshold_d \tag{4.3}$$

and the $p_{i,max}$ will be considered as its class. Based on this definition and using 256,192,128 and 64 as different thresholds, four datasets are acquired. The information about these derived datasets are represented in Table 4.1. Each of these datasets is considered as an independent benchmark to test the proposed method.

4.1.1.1 Cuprite Dataset1

Cuprite dataset1 includes a small number of samples compared to the original Cuprite data. The threshold on the samples in this dataset is 256. It means that in all of these samples the difference of the channel intensity of major class with the other classes in

Table 4.2: Cuprite dataset1 class distribution.

Class	BU	KA	AL
Number	360 (11.74%)	2636 (86%)	69 (2.26 %)

each pixel is 256 or more. These samples are also the most pure of all data sets. This converts the problem to a small size sampling problem (as a classification problem). The data is highly unbalanced. Table 4.2 shows the number of samples per class in this dataset and the Figure 4.3 shows the distribution of the data.

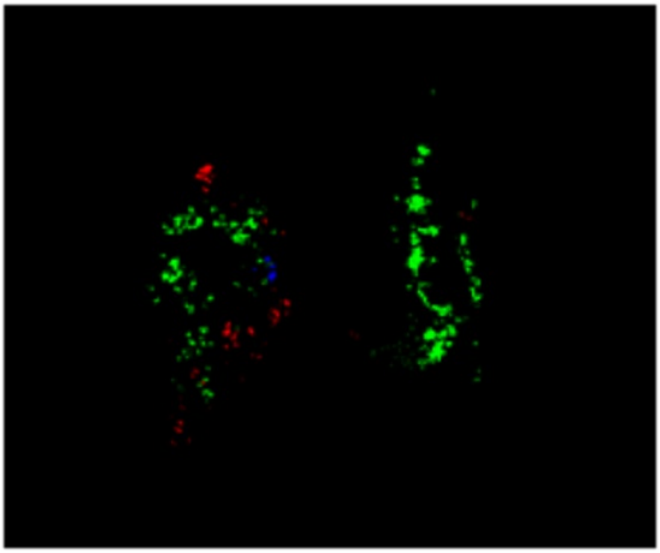


Figure 4.3: Cuprite dataset1 class distribution.

4.1.1.2 Cuprite Dataset2

This dataset includes the samples which their major minerals' intensity are at least 192 larger than the other minerals' intensity. Figure 4.4 shows the distribution of classes in

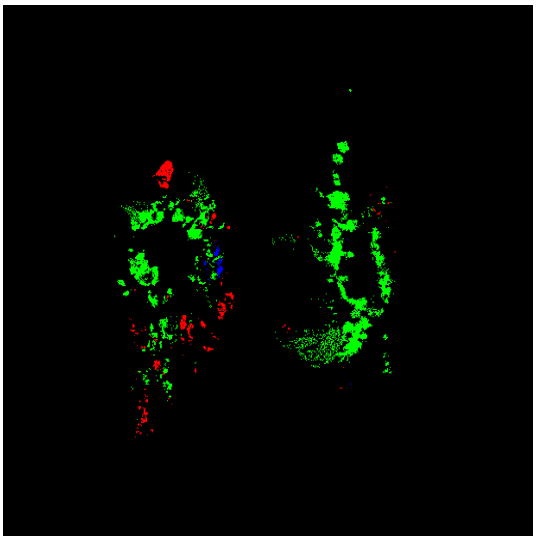


Figure 4.4: Cuprite dataset2 class distribution.

Table 4.3: Cuprite dataset2 class distribution.

Class	BU	KA	AL
Number	972 (12.78%)	6452 (85.44%)	127 (1.68 %)

this dataset. This dataset includes larger number of mixed and noisy samples compared to the previous dataset (Cuprite dataset1). This dataset includes 7,551 samples and the number and percentage of different classes in it is shown in the Table 4.3.

4.1.1.3 Cuprite Dataset3

Cuprite dataset3 consists of the samples which their major minerals' intensity is at least 128 more than the other minerals' intensity. This dataset has 15,850 samples and Table 4.4 presents the number of samples in each class and Figure 4.5 shows the distribution of classes on this dataset.

Table 4.4: Cuprite dataset3 class distribution.

Class	BU	KA	AL
Number	2007 (12.66%)	13435 (84.76%)	418 (2.63 %)

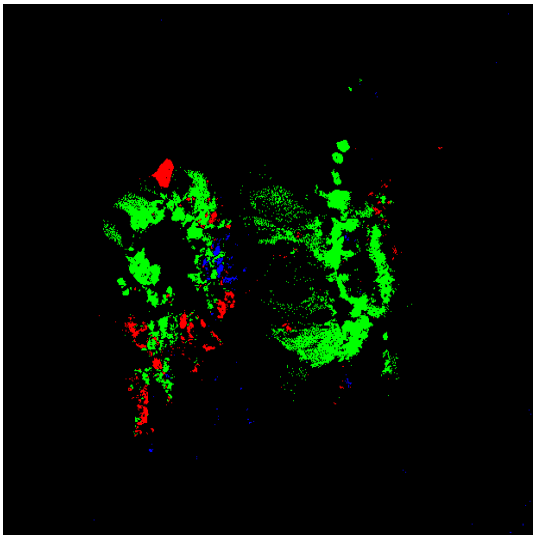


Figure 4.5: Cuprite dataset3 class distribution.

4.1.1.4 Cuprite Dataset4

Cuprite dataset4 includes the samples which their major mineral's intensity in at least 64 more than the other minerals. This dataset includes 81482 samples and Table 4.5 presents the number of samples in each class and Figure 4.6 shows the distribution of classes in this dataset.

Table 4.5: Cuprite dataset4 class distribution.

Class	BU	KA	AL
Number	5346 (6.56%)	28206 (34.61%)	47923 (58.82 %)

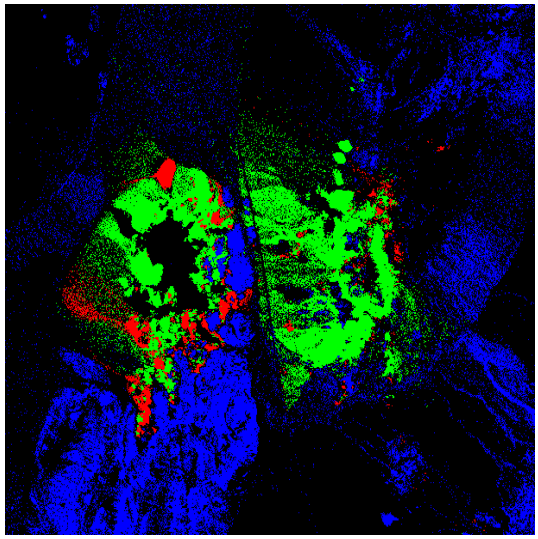


Figure 4.6: Cuprite dataset4 class distribution.

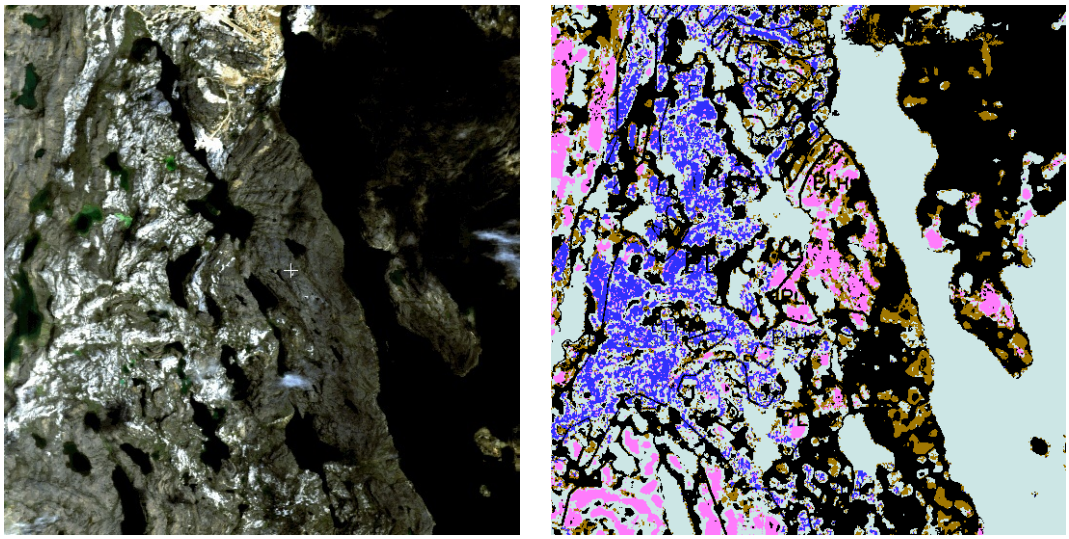
4.1.2 Baffin Island Dataset

Baffin Island Dataset contains hyperspectral data which was obtained from southern Baffin Island, Nunavut in summer of 2004 by the probe-1 hyperspectral sensor. The Probe-1 is a whiskbroom style instrument. It collects data in cross-track direction by mechanical scanning. These data are provided by P. Budkewitsch from the Canada Center for Remote Sensing (CCRS). The acquired data includes bands from wavelengths ranging from 0.433 to 2.513 microns with the ground sampling interval of 0.0165 microns. A total of 127 bands were selected by P. Budkewitsch, while the other bands which contained

primary water/atmosphere noise were removed. Consequently, there are 127 8-bit tiff images each of which demonstrates the absorption and reflection behaviour of the existing materials at a particular spectrum. Each image is considered as a feature dimension. Furthermore, the pixel intensity corresponding to each data sample is its value in that specific feature domain. The corresponding ground truth image for this data set was obtained from the published open file report for that survey [65]. The Baffin Island data is a noisy dataset. The samples in this dataset belong to four major classes. This data has been processed by the Geological Survey of Canada and they classified three rock components (psamite, carbonate and granitoid) [65]. In addition to these three rocks, water is considered as forth component in this experiment because the water is clearly identified on the published map. The main three classes are rocks and composed of mixtures multiple minerals. This means that the data is composed if inherently more complicated spectral mixes than the Cuprite data in which each class consists of an individual mineral, close to their library spectra. The Baffin Island dataset is more realistic than Cuprite data because Cuprite, Nevada is a desert with little vegetation while Baffin Island is rocky, where much of the rock surfaces are covered with lichen, making it inherently more difficult. Table 4.6 shows the number of data samples in each class and Figure 4.7 represents the distribution of classes over the image based. In Figure 4.7 the left image is showing Baffin's RGB image and the right one shows the distribution of classes in it. In the distribution map light blue, brown, dark blue and purple represents water, psammite, carbonate rocks and granitoid, respectively. Black pixels represent unknown samples and are not studied in this experiment.

Table 4.6: Number of different classes.

Class	Water	Psammite (Ps)	Carbonate rocks (Cr)	Granitoid (Gr)
Number	118511	24037	22864	21133



(a) Baffin Island RGB image (b) Baffin class distribution map

Figure 4.7: Baffin data representation. In figure (b) light blue, brown, dark blue and purple represents water, psammite, carbonate rocks and granitoid, respectively.

4.2 Experimental Results and discussion

This section presents the results of applying the algorithm on the datasets discussed in the previous section. The results are reported and the algorithm's performance is discussed.

4.2.1 Classification

In this part of the experimental results section, the results of applying SVM in classifying datasets are presented. In order to find the best results of the SVM the effect of numbers of features and also number of samples for training is studied. Trying different numbers of features helps to evaluate the efficiency of PCA in extracting informative fea-

tures. On the other hand, choosing different training sizes evaluates the functionality of the SVM on obtaining good results based on smaller training sizes.

4.2.1.1 Cuprite Dataset1

In the first step, SVM is applied on the PCA-transformed data. PCA returns the sorted transformed data. Sorting is based on eigenvalues. Therefore, feature selection is also performed. The eigenvalue is an index to show how informative the data is in each dimension. Considering this fact, the number features are important. The implemented classification system tries to use different numbers of features and samples. Each of the reported classification accuracies is the average of 30 runs with the same configuration but different random seeds. Different random seeds resulted in different training and testing sets.

Table 4.7 show the results of applying the classification method on the Cuprite dataset1 test set. The size of training set is indicated in the first row of each column in Table 4.7. This table demonstrates the average performance of multi-SVM classifier system in classifying data with different number of training samples and different number of features over 30 runs. Five SVMs were trained based on class labels generated by ECOC algorithm. In order to evaluate the performance of the classification system on

Table 4.7: Accuracy percentage of SVM on classifying Cuprite Dataset1 test set (average of 30 runs).

Features	303 (10% of each class)	766 (25% of each class)	1532 (50% of each class)	102 (10% of the smallest class)
5	96.03	97.82	**98.39**	97.52
12	74.18	89.03	95.05	98.24
25	33.38	33.95	35.26	41.90

Cuprite dataset1, an accuracy comparison of different SVM configurations is necessary. Table 4.8 summarizes the results of classification with three different feature sets and four different sizes for training sets. The samples of each training set are selected in different ways with different sizes. In the first three columns the number of training sets which have been selected from each class is a percentage of the number of total samples of that class. However in the last column the training set contains equal number of samples for all three classes. The classification results based on using 25 features are obviously not acceptable compared to the other two sets of features (5,12). The set of 5 features showed better performance in three cases out of four. Furthermore, the best performance was obtained by using 5 features and using 50 percent of each class's samples for training.

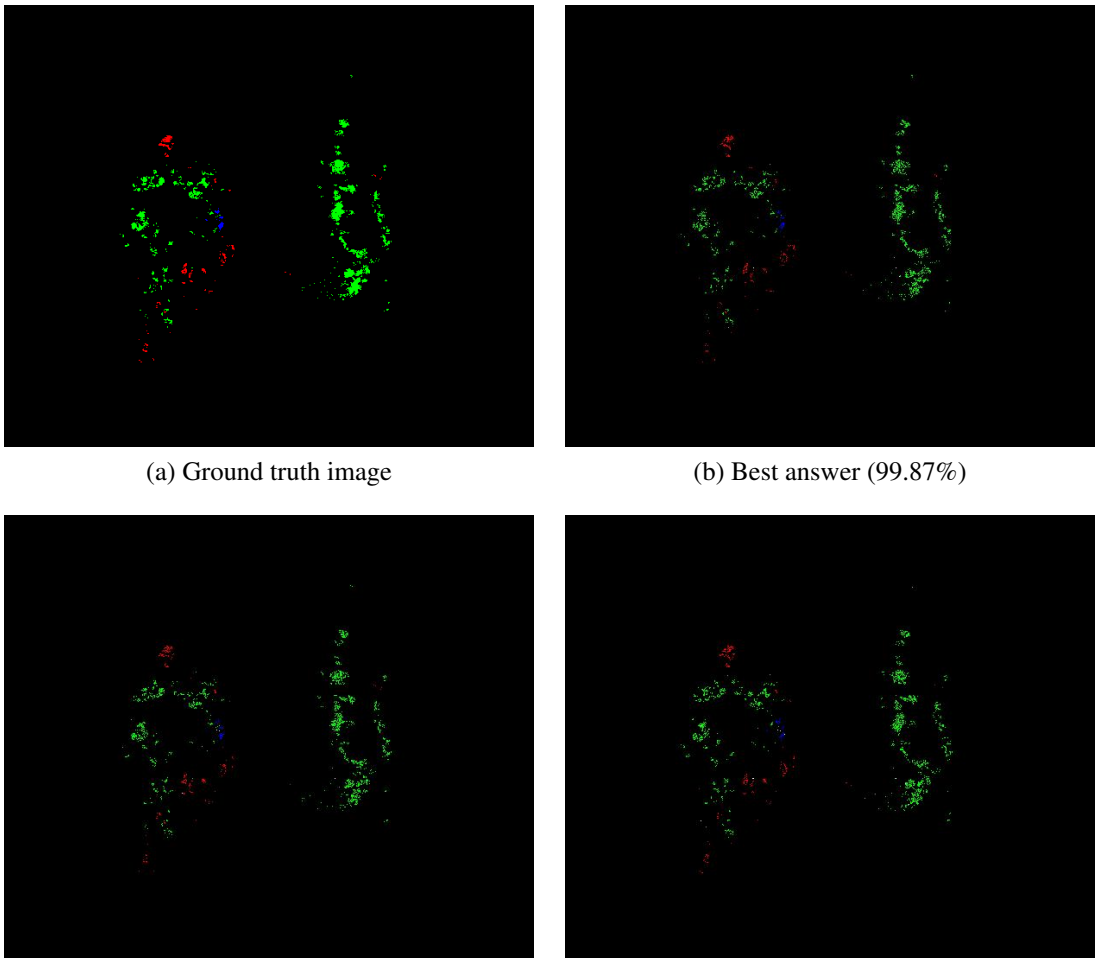
As the results are competitive and the accuracy of different configurations are similars a t-test can determine the significance of one classifier over another. A t-test with a confidence level of 95% has been done on classification results of five best PCA-transferred features and 12 best PCA-transferred features. Table 4.8 show the resulting p-value of each comparison. According to the t-test results, the first three experiments using fine features are significantly better than those using 12 features. Conversely, in the fourth experiment using 12 resulted in better performance than five features. As shown in the table, the best performance is obtained based on a training set that includes 50 percent of the samples and using the first five features (PCA-transformed).

Figure 4.8 visualizes the best, worst and mean runs of this classifier and compares

Table 4.8: Comparison of using different feature sets for classification on Cuprite dataset1 based on t-test with %95 confidence.

features - training size	5-303	5-766	5-1532	5-102
12-303	7E-36 ↑	-	-	-
12-766	-	4.34E-17 ↑	-	-
12-1532	-	-	4.54E-7 ↑	-
12-102	-	-	-	0.008 ←

them with the ground truth image. In this image the red, green and blue colours show pixels corresponding to AL, KA and BU which have been classified correctly. Black colour shows the background. It is worth mentioning again that these pixels (samples) have not been studied in this experiment. The white pixels show the misclassified samples. The results of this experiment is competitive with the other one which was trained on 10 percent of each class and using 12 features for each sample.



(a) Ground truth image

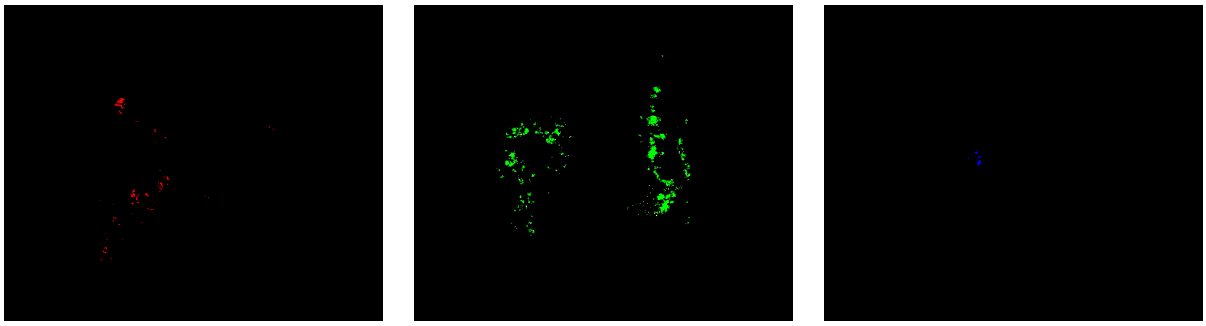
(b) Best answer (99.87%)

(c) Average answer(98.97%)

(d) Worst answer(96.91%)

Figure 4.8: Best, mean and worst performance of best multiSVM configuration on Cuprite dataset1.

Figure 4.9 shows the percentage and distribution of best run of multi-SVM trained based on 1,532 samples and using five PCA-transformed features. As in all the other figures in this chapter, in Figure 4.9 black pixels represent samples which were not involved in this experiment because no classification for these pixels in the answer image. White



(a) Buddingtonite (%99.99) (b) Kaolinite (%99.21) (c) Alunite (99.99%)

Figure 4.9: Best performance of best multiSVM configuration on Cuprite dataset1.

pixels represent classification errors and all other samples are correctly classified.

4.2.1.2 Cuprite Dataset2

Table 4.9 shows the results of applying classification method on PCA-transformed data. Table 4.9 demonstrates the average performance of multi-SVM classifier system

Table 4.9: SVM results on Cuprite Dataset2 test set (average of 30 runs).

Features	754 (10% of each class)	1888 (25% of each class)	3776 (50% of each class)	192 (10% of the smallest class)
5	89.66	93.65	95.04	94.93
12	72.98	87.97	93.62	**96.33**
25	33.35	33.6	34.31	42.60

in classifying data with different number of training samples and features over 30 runs. Five SVMs are trained based on class labels generated by ECOC algorithm. Training classifiers based on 25 features do not yield good results in any of the four training sizes. However, in order to compare the results of using 5 features and 12 features, the t-test

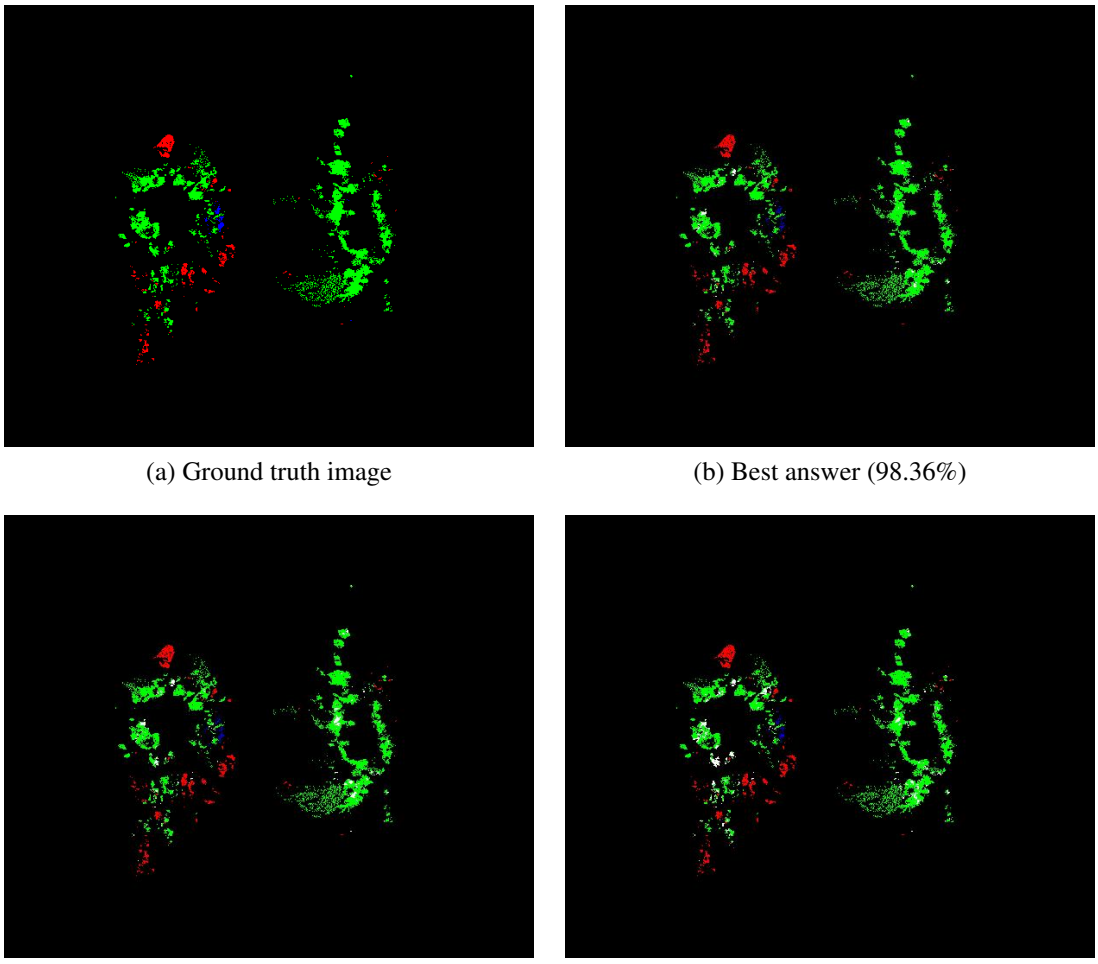
Table 4.10: Comparison of using different feature sets for classification on Cuprite dataset2 based on t-test with %95 confidence.

features - training size	5-754	5-1888	5-3776	5-192
12-754	3.9E-30 ↑	-	-	-
12-1888	-	1.96E-17 ↑	-	-
12-3776	-	-	2.14E-4 ↑	-
12-192	-	-	-	4.96E-8 ←

is needed to test if one is significantly better. Table 5.8 includes the p-value of the t-test with 95% confidence. If one classifier's accuracy is significantly better than the other, an arrow points to the better classifier. For the p-values larger than 0.05 none of the classifier's performance is better than another.

Based on the t-test results which were presented in Table 4.10, the first three experiment using 5 features is significantly better than training the same classifier with 12 features. But in the fourth experiment which the classifiers are trained based on a very small number of data samples, using 12 features is significantly more accurate than using 5 features. Considering that the classifier has been trained using small samples of data, using 5 features is not sufficient and the use of 12 features are more appropriate for these experiments. Conversely, using more features does not improve the accuracy of the results. As it has been shown in the Table 4.9 the best performance is obtained using a training set which consists of 10 percent of the samples and using the first 12 features (PCA-transformed).

Figure 4.10 presents the best, worst and mean run of this classifier and compares them with the answer image. In this image the red, green and blue colours represent pixels corresponding to AL, KA and BU which have been classified correctly. Black colour shows the background and these pixels (samples) have not been studied in this experiment. The white pixels show the misclassified samples. Figure 4.11 shows the percentage and distribution of results of best run multi-SVM classifier. This classifier is



(a) Ground truth image

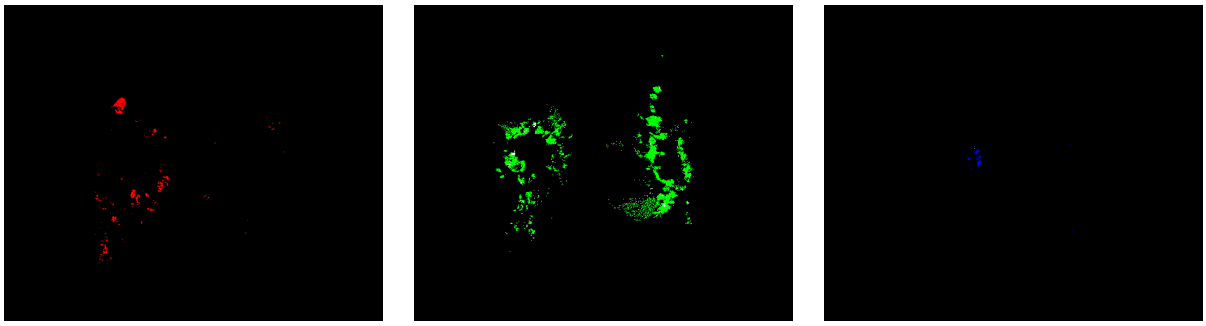
(b) Best answer (98.36%)

(c) Average answer (96.32%)

(d) Worst answer (94.04%)

Figure 4.10: Best, mean and worst performance of best multiSVM configuration on Cuprite dataset2.

trained based of 192 training samples and using the best 12 PCA-transformed features.



(a) Buddingtonite (%100) (b) Kaolinite (%95.83) (c) Alunite (%99.92)

Figure 4.11: Best performance of best multiSVM configuration on Cuprite dataset2.

4.2.1.3 Cuprite Dataset3

The average performance of the classification method over 30 runs has been shown in Table 4.11. The classifier trained on 12 features of 209 samples of each class obtained the best results. This training set included almost 4 percent of all data. Although the overall best answer is obtained by using first 12 PCA-transformed features, however in the first two experiments (1541 training samples and 3966 training samples) the average of the classifier which was trained based on 5 features seems better. In order to evaluate these classifiers precisely, t-test with 95% confidence level is carried out.

Table 4.12 shows the resulting P-values. Considering the p-values presented in Table

Table 4.11: SVM results on Cuprite Dataset3 test set (average of 30 runs).

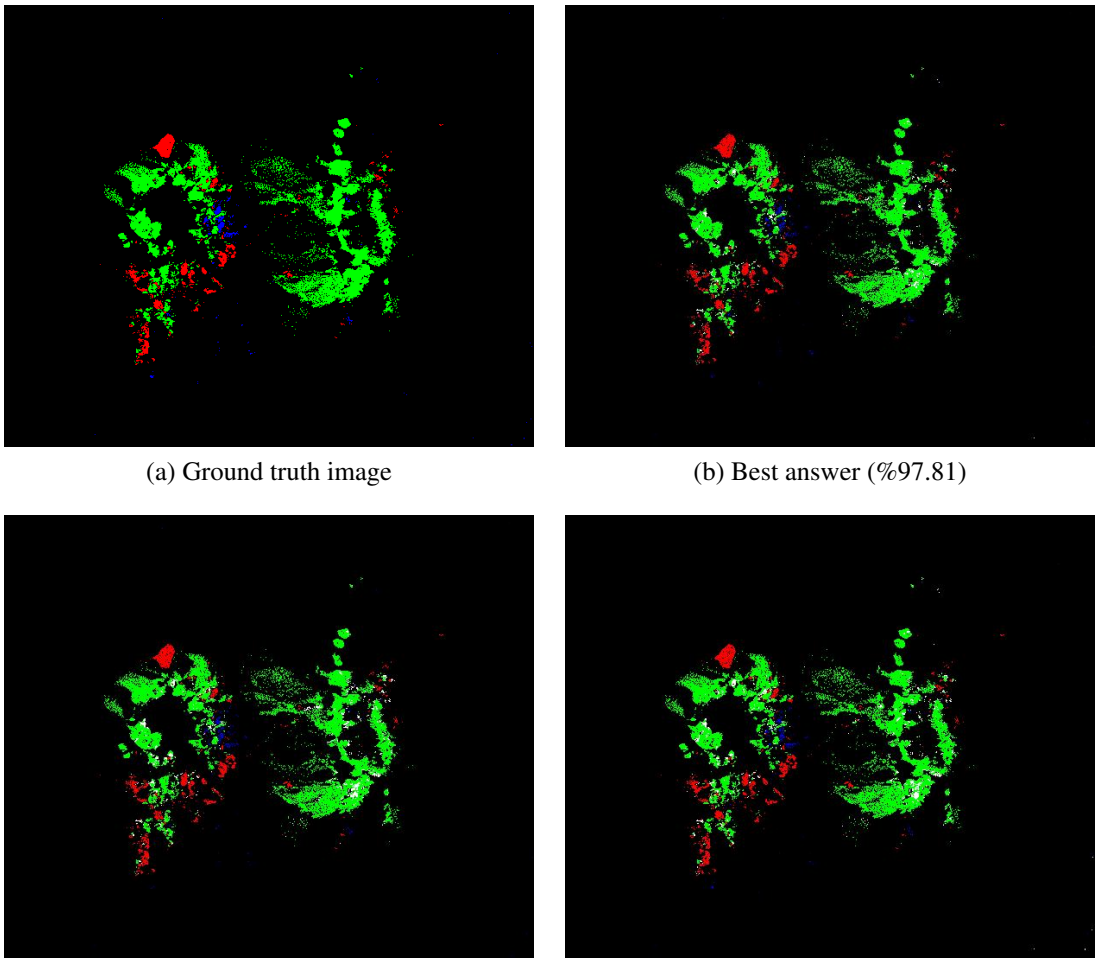
Features	1541 (10% of each class)	3966 (25% of each class)	7931 (50% of each class)	627 (10% of the smallest class)
5	89.51	92.66	94.03	93.63
12	78.07	87.98	93.02	**96.35**
25	33.34	33.70	34.31	47.99

Table 4.12: Comparison of using different feature sets for classification on Cuprite Dataset3 based on t-test with %95 confidence.

features - training size	5-1541	5-3966	5-7931	5-627
12-1541	2.7E-37 ↑	-	-	-
12-3966	-	1.19E-23 ↑	-	-
12-7931	-	-	2.64E-6 ↑	-
12-627	-	-	-	1.77E-19 ←

4.12 using 5 features is significantly better for the unbalanced training sets. Also using 12 features is significantly better for training a classifier with small number of training samples.

The results reported in Table 4.11 are the average performance of the classification system over 30 runs. The output of the best, mean and worst run of the best configuration (209 training samples from each class and using the best 12 PCA-transformed features) is demonstrated in the Figure 4.12.



(a) Ground truth image

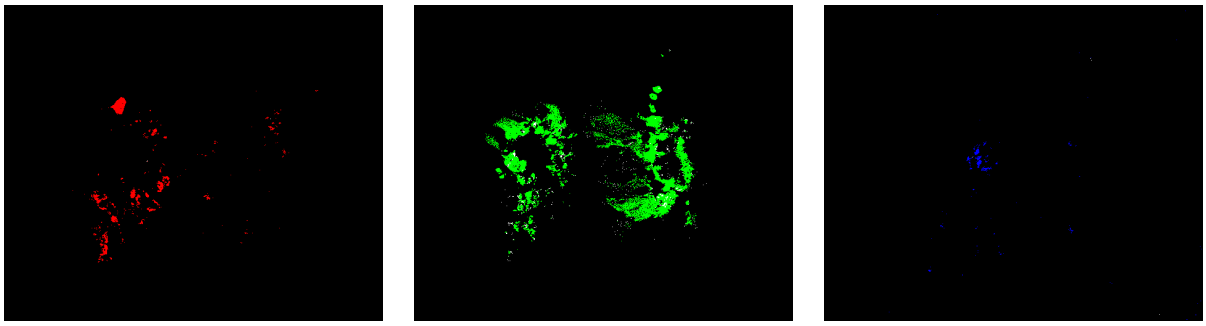
(b) Best answer (%97.81)

(c) Average answer (%92.38)

(d) Worst answer (%91.89)

Figure 4.12: Best, mean and worst performance of best multiSVM configuration on Cuprite dataset3.

Figure 4.13 shows the distribution (and accuracy percentage) of results of best run multi-SVM classifier on Cuprite Dataset3 test set. this classifier is trained based of 627 training samples and using the best 12 PCA-transformed features.



(a) Buddingtonite (%99.91) (b) Kaolinite (%94.83) (c) Alunite (%99.9)

Figure 4.13: Best performance of best multiSVM configuration on Cuprite dataset3.

4.2.1.4 Cuprite Dataset4

The average performance of the classification method over 30 runs has been shown in Table 4.13. Table 4.14 includes the p-values resulting from applying t-test on the results

Table 4.13: SVM results on Cuprite Dataset4 test set (average of 30 runs).

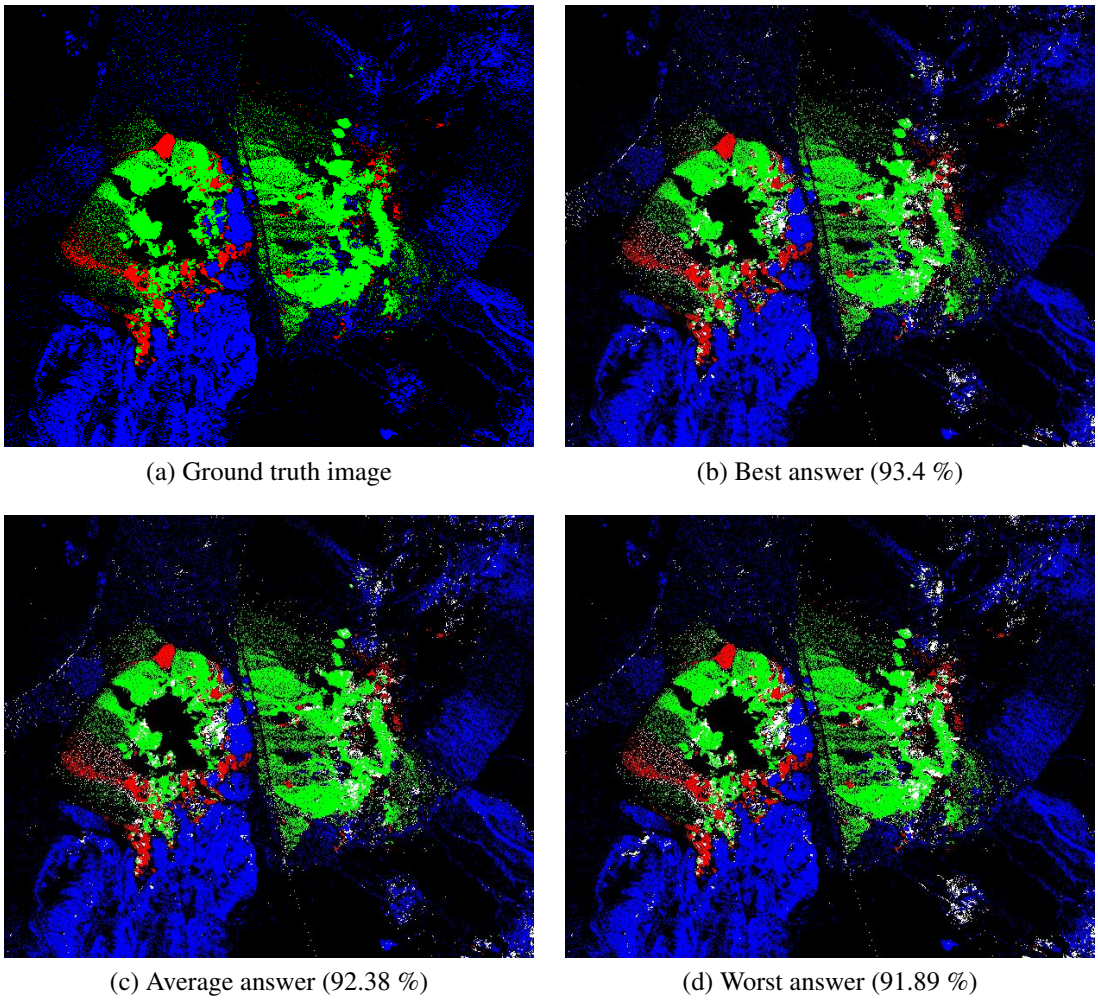
Features	8148 (10% of each class)	20371 (25% of each class)	40742 (50% of each class)	1602 (10% of the smallest class)
5	88.37	93.38	93.76	93.34
12	84.14	94.84	95.06	**95.53**
25	33.38	73.65	77.34	57.52

of using 5 features and 12 features in each experiment. It is necessary to check if one of them showed a significantly better performance compared to the other classifier. Table 4.14 indicates whether in each experiment there is a classifier significantly better than another (with 95% confidence). For first 3 experiments 5 features is significantly better than 12 features. However for classifiers which use small number of samples for training, 12 features is significantly better. The classifier which was trained on 534 samples of each

Table 4.14: Comparison of using different feature sets for classification on Cuprite dataset4 based on t-test with %95 confidence.

features - training size	5-8148	5-20371	5-40742	5-1602
12-8148	5.9E-39 ↑	-	-	-
12-20371	-	5.5E-18 ↑	-	-
12-40742	-	-	2.8E-41 ↑	-
12-1602	-	-	-	4.2E-40 ←

class demonstrated the best results. This classifier used the best 12 features of data. The best, mean and worst performance of this classifier have been compared with the answer image in Figure 4.14.



(a) Ground truth image

(b) Best answer (93.4 %)

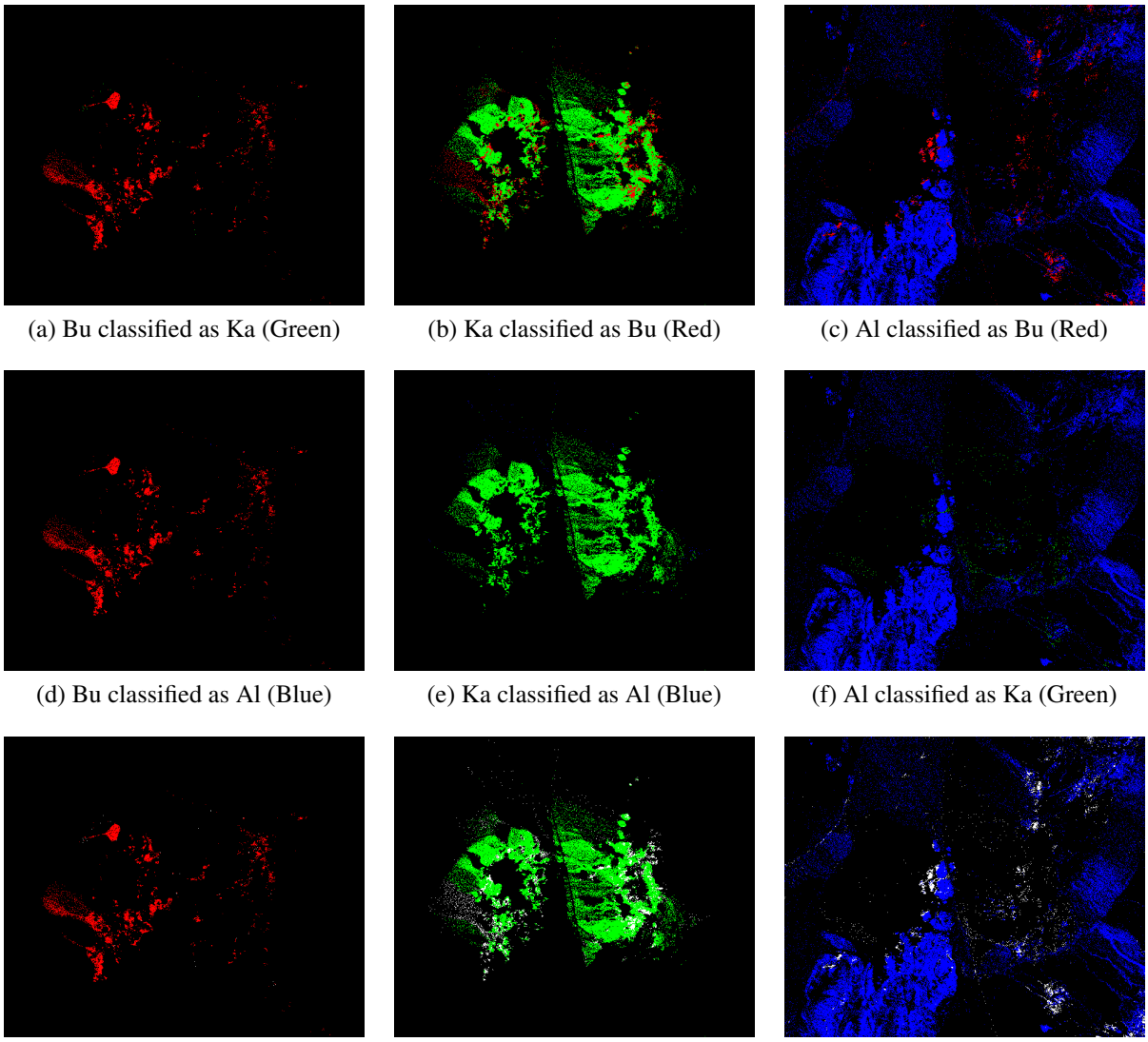
(c) Average answer (92.38 %)

(d) Worst answer (91.89 %)

Figure 4.14: Best, mean and worst performance of best multiSVM configuration on Cuprite dataset4.

Figure 4.15 shows the distribution (and accuracy percentage) of the results of the best run of the multi-SVM classifier on Cuprite Dataset4 test set. Each column shows the results of classifier on one class. The first column of images is dedicated to showing the accuracy of best SVM run on classifying buddingtonite. In this column red pixels are the

buddingtonite samples which are correctly classified. In the first image, the pixels which belong to buddingtonite but classified as kaolinite are differentiated with green colour. In the second image of this column, the buddingtonate samples which are misclassified as alunite are differentiated with blue colour. And the last image of this column colours all misclassified samples (of buddingtonite) with white. The other two columns show misclassified samples of kaolinite and alunite in the same format. As it is clear in the second column, almost all of the kaolinites misclassified samples are classified as buddingtonite, and similarly for alunite. As it can be seen it the third column there are not a lot of alunite samples that have been classified as kaolinite and the major alunies misclassified samples are classified as buddingtonite. This classifier is clearly biased towards buddingtonite.



(a) Bu classified as Ka (Green) (b) Ka classified as Bu (Red) (c) Al classified as Bu (Red)

(d) Bu classified as Al (Blue) (e) Ka classified as Al (Blue) (f) Al classified as Ka (Green)

(g) Buddingtonite (99.18%) (h) Kaolinite (86.73%) (i) Alunite (95.71%)

Figure 4.15: Best performance of best multiSVM configuration on Cuprite dataset4.

4.2.1.5 Baffin Island Dataset

PCA-transformed data is used for classification. In classification process 7 SVMs have been trained to classify 4 classes. The binary class labels are generated by ECOC. Table 4.15 shows the binary label of each class for each SVM. The set of SVMs which

Table 4.15: Binary codes generated by ECOC for multiSVM system (4 classes).

Class	**SVMs**						
1	1	1	1	0	1	1	0
2	1	0	0	1	0	1	0
3	0	0	1	1	1	1	1
4	1	1	0	1	1	0	1

were trained based on this coding classified the data. The accuracy of the SVMs trained on different numbers of samples and using different numbers of features are presented in Table 4.16. Table 4.17 includes the p-values resulting from applying t-test (with 95% confidence) on the results of classifiers trained on 5 and 12 PCA-transformed features. According to Table 4.17 the first two experiments using 5 features are significantly better and the fourth experiment (using 7044 samples per each class) using 12 features is significantly better than 5 features and 25 features. Classifiers which were trained based on 2817 samples per each class. Using 12 features or 5 features is not making any significant

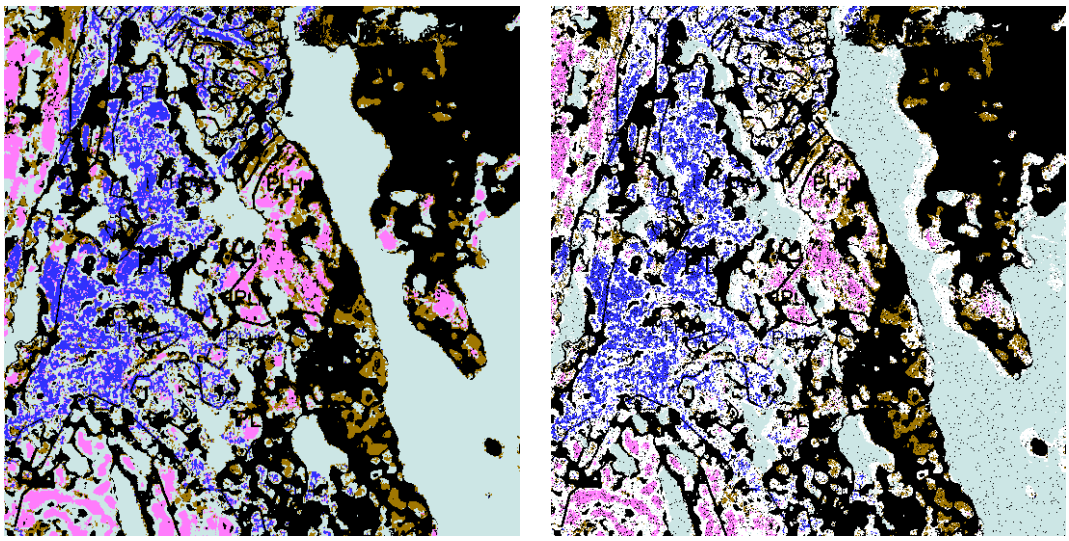
Table 4.16: SVM results on Baffin Island test set (average of 10 runs).

Features	18653 (10% of each class)	46598 (25% of each class)	8452 (10% of the smallest class)	21132 (25% of the smallest class)
5	53.49	54.69	55.35	56.28
12	49.02	52.06	55.70	**60.71**
25	25.60	26.72	36.94	41.44

Table 4.17: Comparison of using different feature sets for classification on Baffin Island dataset based on t-test with %95 confidence.

features - training size	5-18653	5-46598	5-8452	5-21132
12-18653	8.31E-33 ↑	-	-	-
12-46598	-	2.49E-32 ↑	-	-
12-8452	-	-	0.066	-
12-21132	-	-	-	1.3E-13 ←

difference. The best results are acquired by the multi-SVM system which used the 12 best features provided by PCA and used only 25 percent of samples of each class for training. Figure 4.16 represents the accuracy of this classifier. The white pixels show misclassified samples.



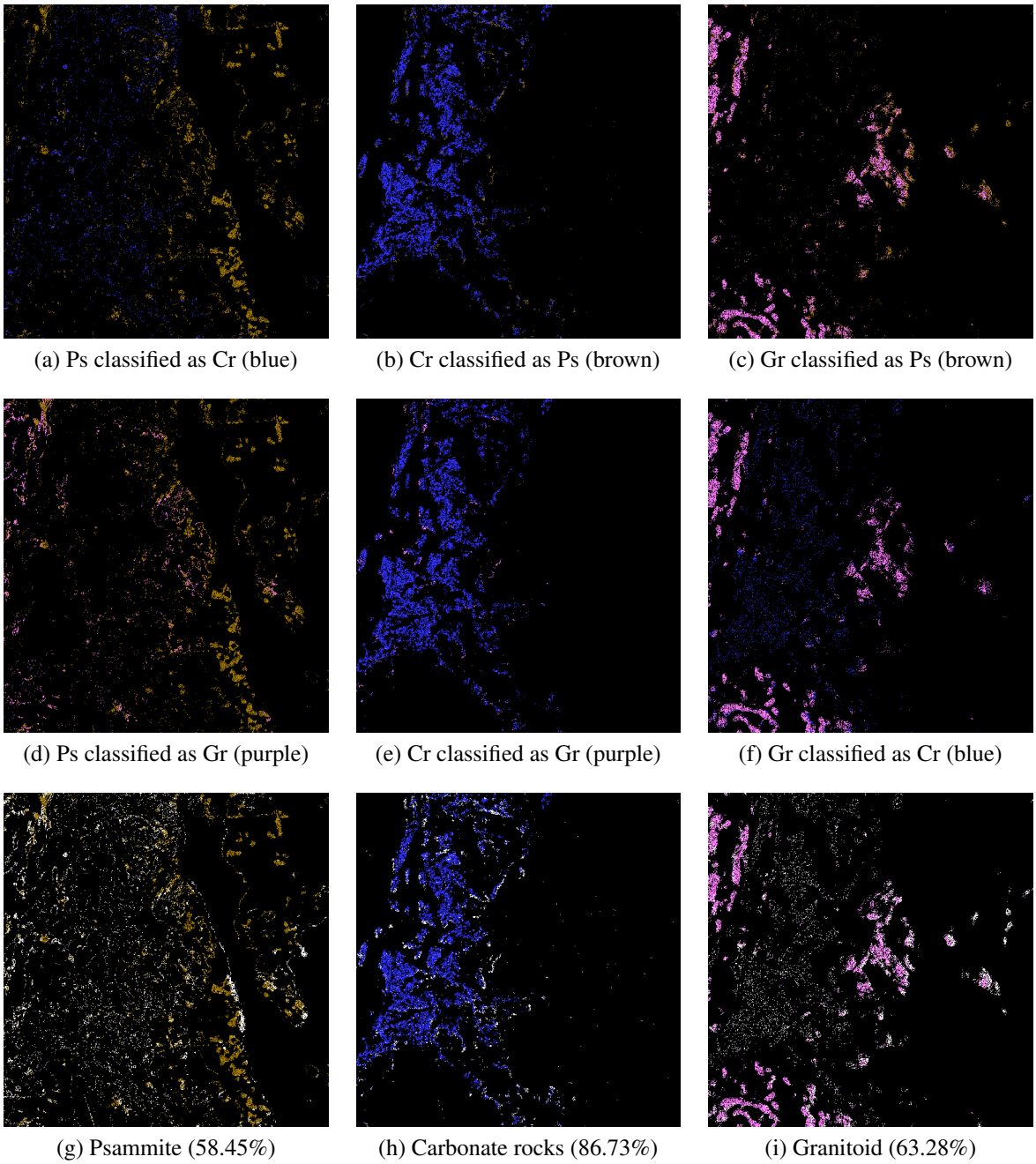
(a) Ground truth image (b) Best classification results

Figure 4.16: Representation of the best SVM results on Baffin data.

Figure 4.17 shows the distribution (and accuracy percentage) of results of the multi-

SVM classifier (the best run) on Baffin Island test set. Each column shows the results of classifier on one rock. The images of first column belong to psammite. In the first image of this column, pixels which belong to psammite class but are classified as carbonate are coloured blue. The first two images of each column show the pixels misclassified as another rock and the last image (of each column) shows the result of classification of each rock's pixels. In the images in the third row white pixels represent misclassified pixels of each rock. As water could be removed by thresholding before classification, the distribution of errors over water is not presented in this figure. This classifier is trained based of 21132 training samples and using the best 12 PCA-transformed features.

The first column of Figure 4.17 presents the errors of the classifier on classifying psammite samples. Most of the misclassified psammite samples are classified as carbonate rocks. The second column of images shows that the classifier classified carbonate rocks more accurate rather than the other two classes. As the first two images of the last column are showing, most of the misclassified granitoid samples are classified as carbonate rocks. Accounting to the images in 4.17 this classifier is based towards the carbonate rocks. As a result, the accuracy of classifying carbonate rocks is significantly higher than the other classifiers. Also most of the misclassified samples of psamite and granitoid are classified as carbonate rocks.



(a) Ps classified as Cr (blue) (b) Cr classified as Ps (brown) (c) Gr classified as Ps (brown)

(d) Ps classified as Gr (purple) (e) Cr classified as Gr (purple) (f) Gr classified as Cr (blue)

(g) Psammite (58.45%) (h) Carbonate rocks (86.73%) (i) Granitoid (63.28%)

Figure 4.17: Results of best multi-SVM configuration on Baffin Island testing data.

4.2.2 Clustering

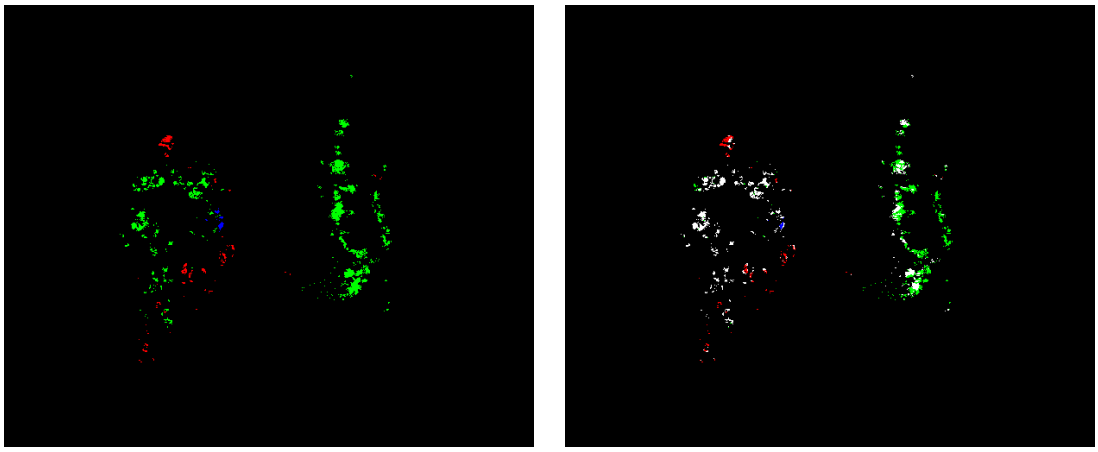
In this section the performance of the selected clustering algorithm is evaluated. The effect of using different numbers of features on the performance of self-organizing maps is examined. Another important variable is the number of initial nodes on the SOM as in many cases the number of existing clusters are not specified. Similar to the classification experiments, the black pixels are representing the samples that are not included in the experiment.

4.2.2.1 Cuprite Dataset1

Table 4.18 contains the results of clustering Cuprite Dataset1 with different configurations. In Table 4.18 each column shows the effect of different number of features on performance of an SOM with a constant structure. In the first three rows the SOMs used different subsets of PCA-transformed features and the last row shows the results of using all features without applying any feature extraction methods. According to this table selecting various sets of features does not result in significant changes. However, the number of initial nodes is important. Figure 4.20 shows the areas and pixels categorized correctly and compares the results of the SOM with the answer map. In the image the coloured areas are the ones which are correctly clustered.

Table 4.18: Self-organizing maps results on Cuprite Dataset1.

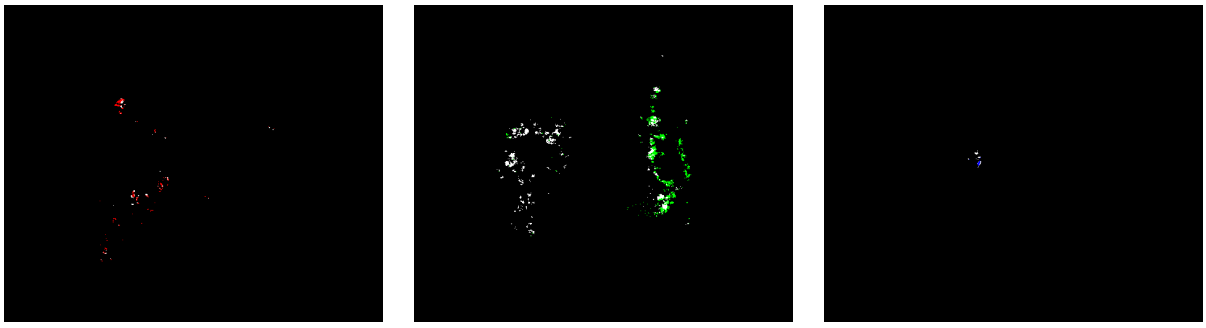
Features	**Initial Nodes**		
	3	5	9
5 PCA	**51.27**	47.6	30.37
12 PCA	50.72	47.53	30.64
168 PCA	50.77	47.76	30.85
168 non-PCA	50.85	47.77	30.9



(a) Ground truth image (b) Best Self-organizing map

Figure 4.18: Performance of best self-organizing maps on Cuprite Dataset1.

Figure 4.19 represents the performance accuracy and the distribution of clustering Cuprite Dataset1 by SOM. In this experiment which gained the best results, SOM uses the best 5 PCA-transformed data.



(a) Buddingtonite (70.69%) (b) Kaolinite (5284%) (c) Alunite (46.33%)

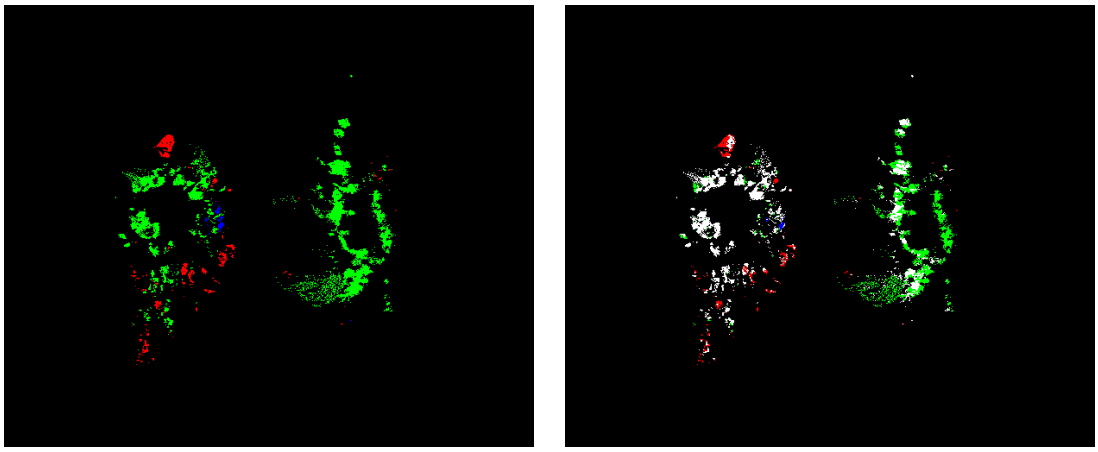
Figure 4.19: Performance of the best SOM configuration on Cuprite Dataset1.

4.2.2.2 Cuprite Dataset2

Here, the SOM's ability in unsupervised categorizing is evaluated on Cuprite Dataset2. Table 4.19 presents the results of SOM with different number of feature and different number of initial nodes. According to Table 4.19 the accuracy of SOM for this dataset depends on the number of initial nodes. Conversely, the number of features does not help much to improve the results. Similar to the Cuprite Dataset1 the clustering results based on non-transformed features are worse compared to the transformed data. Figure 4.20 represents the performance of SOM with the best configuration on the Cuprite Dataset2.

Table 4.19: Self-organizing maps results on Cuprite Dataset2.

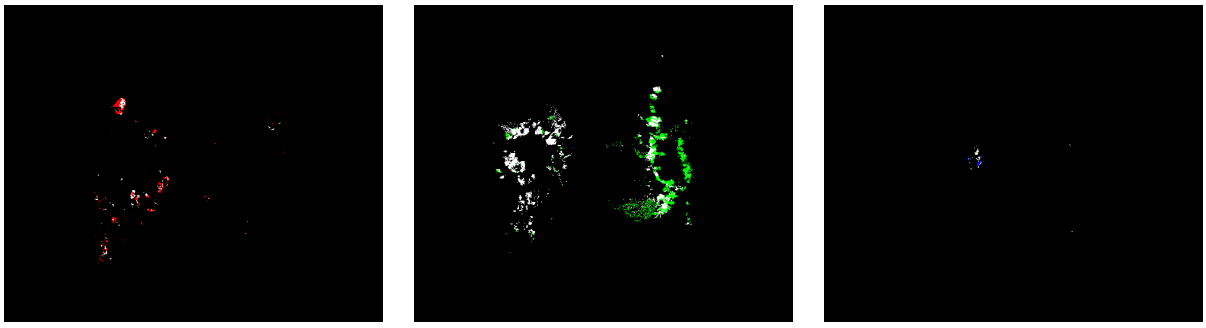
Features	**Initial Nodes**		
	3	5	9
5 PCA	**47.44**	40.73	25.10
12 PCA	47.40	40.90	25.30
168 PCA	47.40	40.65	25.38
168 non-PCA	33.02	32.42	24



(a) Answer map (b) Best Self-organizing map

Figure 4.20: Performance of best self-organizing maps on Cuprite Dataset2.

Figure 4.21 represents the performance accuracy and the distribution of clustering Cuprite Dataset2 by SOM. In this experiment which gained the best results, SOM uses the best 5 PCA-transformed data.



(a) Buddingtonite (61.29%) (b) Kaolinite (46.76%) (c) Alunite (38.62%)

Figure 4.21: Performance of the best SOM configuration on Cuprite Dataset2.

4.2.2.3 Cuprite Dataset3

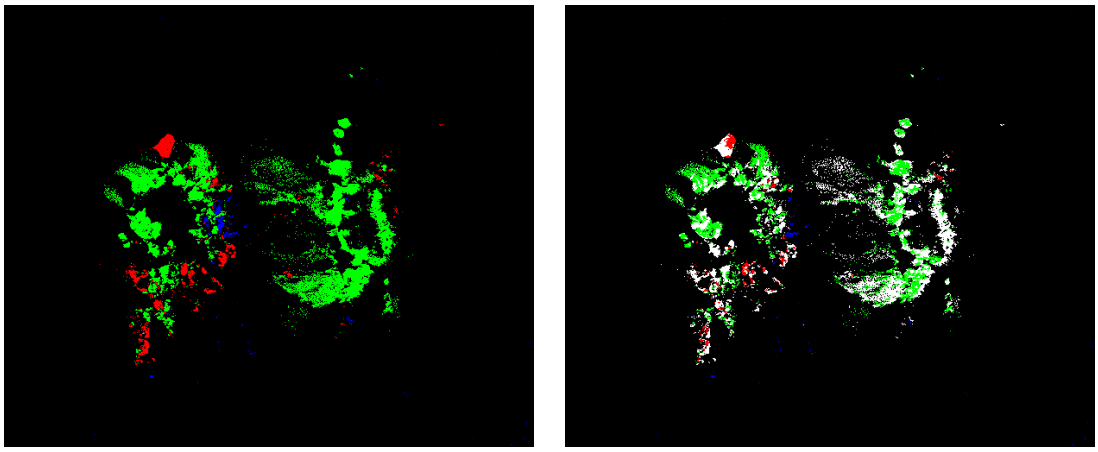
To evaluate the SOM's ability to learn this data, the SOM is applied to Cuprite Dataset3 with different configurations. These differences are in two main areas: using various feature sets and also different numbers of initial nodes. Three sets of PCA-transformed data as well as all features of non-transformed data have been considered as different feature sets. This experiment also evaluated the usability of PCA feature extraction. Different numbers of initial nodes have been used to determine if SOM is able to categorize the data without knowing the actual number of groups. Table 4.20 shows the results of different SOM configurations on Cuprite Dataset3. According to the Table

Table 4.20: Self-organizing maps results on Cuprite Dataset3.

Features	**Initial Nodes**		
	3	5	9
5 PCA	**41.06**	36.72	25.27
12 PCA	41.05	36.74	25.95
168 PCA	41.04	36.70	26.10
168 non-PCA	33.02	32.42	24

4.20, SOM's accuracy is highly sensitive to the number of initial nodes. By changing the number of nodes from 3 to 5 the accuracy of SOM is reduced around 5%. On the other hand, the PCA transformed data positively affected the SOM's results. Using different numbers of PCA-transformed features didn't make any difference in clustering accuracy. However, non-transformed features showed lower performance, regardless of number of initial nodes. Although the advantage of using PCA-transformed data is more sensible in SOM with 3 initial nodes.

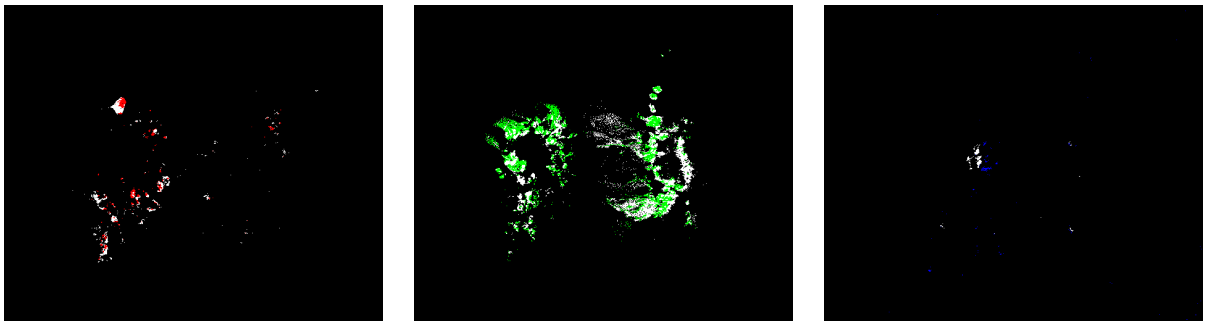
Figure 4.22 compares the obtained results of the SOM with 3 initial nodes and used 5 features which showed the better performance compared to the other configurations.



(a) Answer map (b) Best Self-organizing map

Figure 4.22: Performance of best SOM on Cuprite Dataset3.

Figure 4.23 represents the performance accuracy and the distribution of clustering Cuprite Dataset3 by SOM with the same configuration as the one in Figure 4.22. Figure 4.23 presents the performance of SOM on each data class separately. As it is shown in this figure, SOM obtained its best results on clustering kaolinite and alunite appeared as the hardest class for SOM to be categorized.



(a) Buddingtonite (36.64%) (b) Kaolinite (45.59%) (c) Alunite (23.18%)

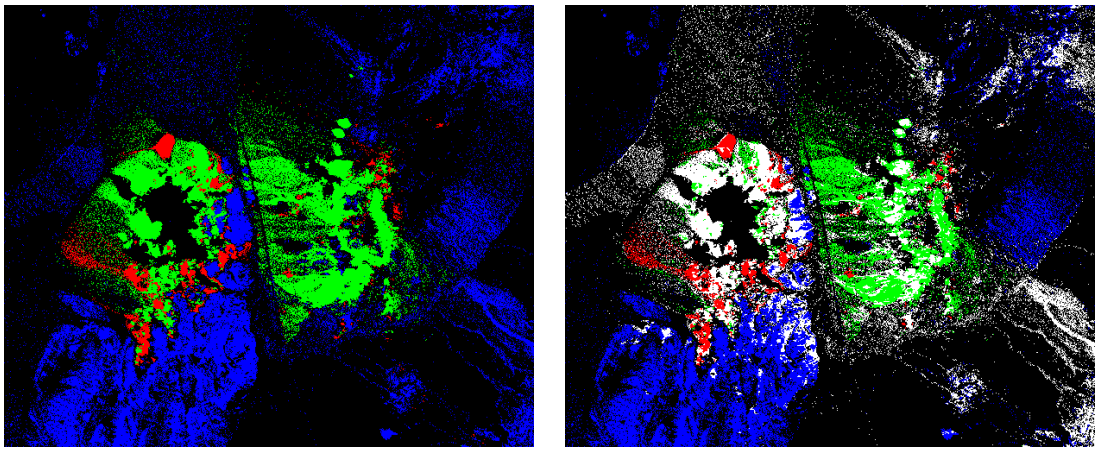
Figure 4.23: Performance of the best SOM configuration on Cuprite Dataset3.

4.2.2.4 Cuprite Dataset4

Table 4.18 represents the results of SOM's different configurations Cuprite Dataset4. According to the results of different configurations of SOM in categorizing Cuprite Dataset4, the most important factor in SOM's accuracy is the number of initial nodes. The SOM with 3 initial nodes (the same as class numbers) shows significantly better results. The next important factor is feature extraction. In the SOM with 3 initial nodes, PCA-transformed data showed significantly better results comparing to non-transformed data. It also concluded that using different numbers of PCA-transformed data does not affect SOM accuracy on clustering Cuprite datraset4. Figure 4.24 shows the results of the SOM with 3 initial nodes which used all 168 PCA-transformed features (best SOM configuration).

Table 4.21: Self-organizing maps results on Cuprite Dataset4.

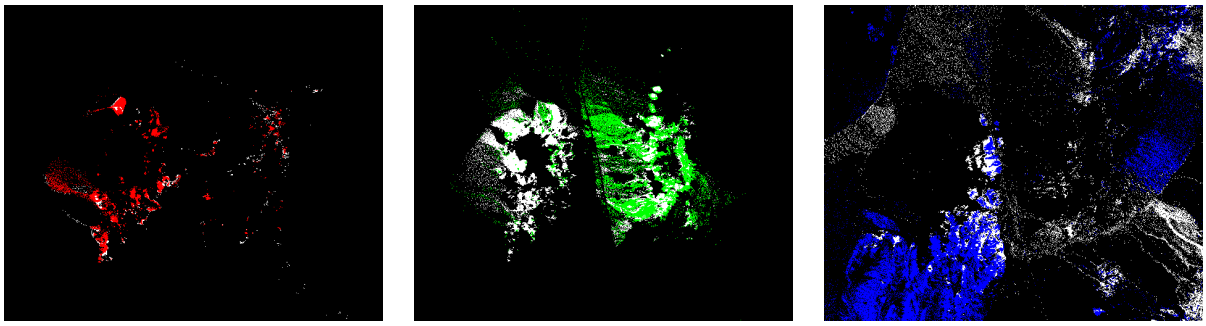
Features	**Initial Nodes**		
	3	5	9
5 PCA	63.35	33.75	33.14
12 PCA	63.35	33.78	33.22
168 PCA	**63.36**	33.86	33.30
168 non-PCA	55.17	43.03	26.06



(a) Answer map (b) Best Self-organizing map

Figure 4.24: Performance of best self-organizing maps on Cuprite Dataset4.

Figure 4.25 represents the performance accuracy and the distribution of clustering Cuprite Dataset4 by SOM with the same configuration as the on in 4.24. Figure 4.25 presents the performance of SOM on each data class separately. SOM categorized buddingtonate significantly better than other two classes. Alunite appeared as the hardest class of Cuprite dataset4 to be identified.



(a) Buddingtonite (81.71%) (b) Kaolinite (46.90%) (c) Alunite (61.51 %)

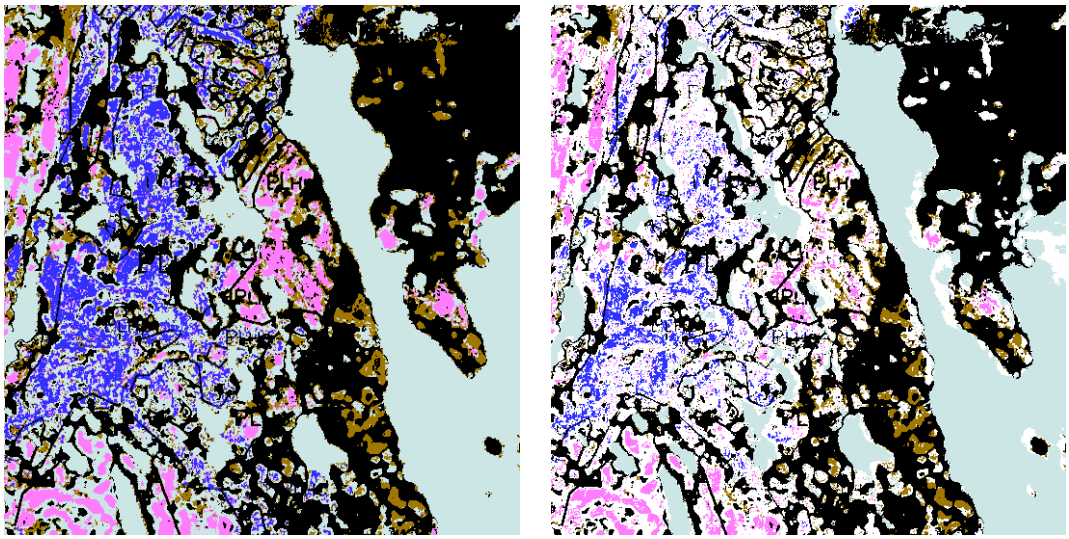
Figure 4.25: Performance of the best SOM configuration on Cuprite Dataset4.

4.2.2.5 Baffin Island Dataset

SOMs also have been applied on Baffin data. Different numbers of initial nodes and various sets of features are considered in the experiment to find the best SOM configuration. The ability of SOMs to identify the data without having information about the number of categories is examined. Table 4.22 shows the results of different SOM configurations on Baffin data. Figure 4.26 represents the accuracy of the best SOM which used the best 12 features provided by PCA and 4 initial nodes.

Table 4.22: Self-organizing maps results on Baffin data.

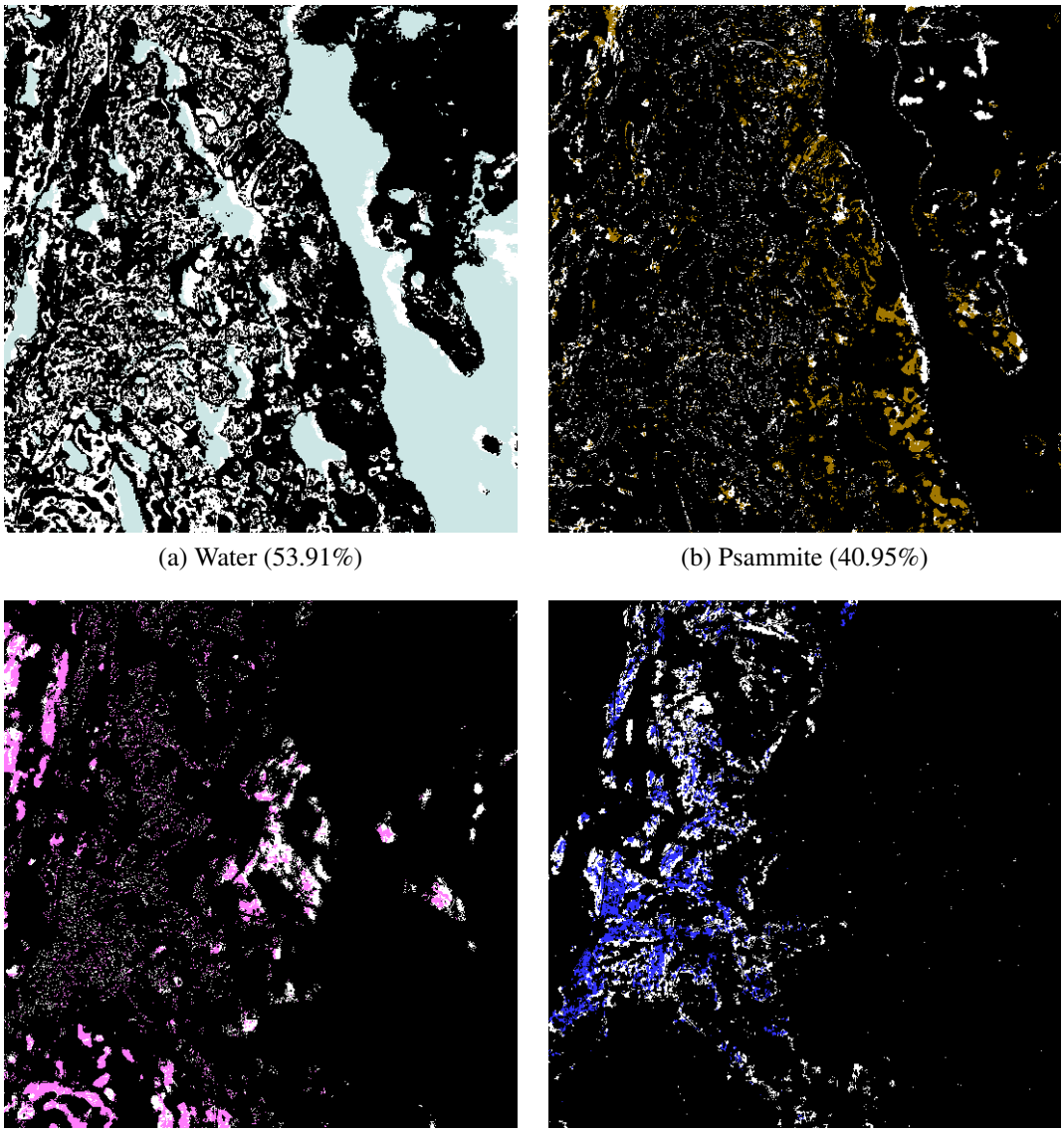
Features	**Initial Nodes**		
	4	6	9
5 PCA	46.79	36.68	26.59
12 PCA	**46.82**	36.48	26.57
168 PCA	46.80	36.47	28.61
168 non-PCA	46.81	36.90	28.59



(a) Baffin answer map (b) best SOM results

Figure 4.26: Respresentation of the best SOM results on Baffin data.

Figure 4.27 shows accuracy with the best configuration on each class separately. It also demonstrates the distribution of mis-clustered samples. SOM showed the best results in categorizing water and granitoid rocks and the worst results on carbonate rocks. In case of water, the SOM successfully identified the samples of large body of water, which are very similar. But the small areas of water are not identified. The distribution of clustering error on psammite and carbonate rocks are also similar. The scattered samples which are the noisy ones are mostly missed in clustering process. A large portion of granitoid rocks' errors are located in the middle of image.



(a) Water (53.91%)

(b) Psammite (40.95%)

(c) Granitoid roks (55.92%)

(d) Carbonate rocks (35.69%)

Figure 4.27: Performance of best SOM configuration on Baffin data.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This thesis presents both supervised learning and unsupervised learning techniques on hyperspectral remote-sensing data. Another subject which was discussed in this thesis is the effect of principal component analysis in reducing the noise and useless features. According to the results of classification and clustering methods, PCA increased the accuracy of learning. The best operation of learning methods was obtained with a small number of transformed features. In the classification process performed by multi-SVM systems, results for Cuprite data show that increasing the number of included samples adds more noise to the data. The number of features needed to gain the best result increases from 5 to 12. Although the first five transformed features by PCA contain the most informative feature spaces, that information was not sufficient to separate different classes. On the other hand, adding more features deteriorated the SVMs performance in separating the classes. The reason is that classification on spaces with data that are not separable, forces SVMs to make more exceptions in order to be able to converge to a separator. This effect reduces the generalization ability of SVMs.

Using error-correction output codes (ECOCs) helped to correctly classify the samples which have been misclassified by just one classifier. The one advantage of this multi-class SVM strategy compared to the other strategies like "one against one and "one against all. Using the appropriate PCA-transformed data improved the accuracy of SVMs. It also reduced the processing time, which is very important. One of the disadvantages of an SVM kernel like QP is that it is very time consuming. Reducing the dimensions of data is one way to make QP feasible. One important difference between the Cuprite data set and the Baffin Island data is that the ground truth image is essentially a binary image: e.g. a pixel is classified as a psammite or not. Hence no experiments based on the purity of a pixel can be performed. This fact could be one of the reasons that SVM showed significantly better performance on Cuprite data compared to the Baffin Island data.

The self-organizing maps showed noticeably better performance on the Baffin data. Considering the fact that during the classification process a portion of samples are used for training and that class labels are needed for the data, clustering appears to be more independent and general. However, for the same reason, the expectation for the results of clustering are normally lower than for classification methods. In Baffin Island data, which is a hard case for classification, SOM shows good results and challenged the SVMs performance. Although it should be noted that the accuracy of SOM is dependent to the number of initial nodes more than other variables. This fact questions the ability of SOM to perform properly without having at least an idea of the number of categories of the data. As the cost of labeling a part of data (training set) is considerable, SOM could be used an acceptable alternative for classification. Although choosing a different number of features did not make considerable improvement in the clustering process, SOM performed better with PCA-transformed data compared to regular data on the Cuprite dataset. Although choosing different sets of features or transforming the data did not improve the SOMs performance significantly. SOM did not succeed in clustering with the wrong number of initial nodes. It means that SOM works well if the approximate number of categories is known.

5.2 Future Work

There are many possible directions of future research in hyperspectral remote sensing for mineral identification. Noise and large number of features are the natural properties of the problems in this field of study. For future work there are many aspects of our algorithm that could be improved:

- Feature extraction: Applying new feature extraction methods to unmix the samples. By reducing the general effect of noise, extracting the initial vectors (mineral's feature vectors) is possible. Linear discriminant analysis is a successful feature extractor for binary-class problems.

- Feature selection: PCA by returning eigenvalues gives an index to select best features. Several feature selection methods could be applied. The genetic algorithm is one popular method which could be employed for this problem.

- Clustering: SOM only succeeded in clustering if the original number of categories was approximately known. K-nearest neighbour is another clustering technique. It is not able to estimate the number of clusters. Rather, it is a powerful clustering algorithm.

Bibliography

[1] Alisneaky. Kernel machines are used to compute a non-linearly seperable functions into a higher dimension linearly seperable function. `http://en.wikipedia.org/wiki/File:Kernel_Machine.png`, April, 2011.

[2] Yakoub Bazi and Farid Melgani. Toward an optimal svm classification system for hyperspectral remote sensing images. *Geoscience and Remote Sensing, IEEE Transactions on*, 44(11):3374–3385, 2006.

[3] Richard Ernest Bellman. Some new techniques in the dynamic-programming solution of variational problems. pages 1148–1169, 1957.

[4] Richard Ernest Bellman and Bellman. *Adaptive control processes: a guided tour*. Princeton University Press, 1966.

[5] Charles L Bennett, Michael R Carter, David J Fields, and F Dean Lee. Infrared hyperspectral imaging results from vapor plume experiments. In *SPIE's 1995 Symposium on OE/Aerospace Sensing and Dual Use Photonics*, pages 435–444. International Society for Optics and Photonics, 1995.

[6] Irving Biederman et al. Recognition-by-components: A theory of human image understanding. *Psychological review*, 94(2):115–147, 1987.

[7] José M Bioucas-Dias, Antonio Plaza, Nicolas Dobigeon, Mario Parente, Qian Du, Paul Gader, and Jocelyn Chanussot. Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches. *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, 5(2):354–379, 2012.

[8] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 1. Springer, 2006.

[9] Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In *Advanced Lectures on Machine Learning*, pages 169–207. Springer, 2004.

[10] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

[11] David M Cairns. A comparison of methods for predicting vegetation type. *Plant Ecology*, 156(1):3–18, 2001.

[12] Gustavo Camps-Valls and Lorenzo Bruzzone. Kernel-based methods for hyperspectral image classification. *Geoscience and Remote Sensing, IEEE Transactions on*, 43(6):1351–1362, 2005.

[13] Chein-I Chang. *Hyperspectral imaging: techniques for spectral detection and classification*. Springer, 2003.

[14] Chien-I Chang and Qian Du. Interference and noise-adjusted principal components analysis. *Geoscience and Remote Sensing, IEEE Transactions on*, 37(5):2387–2396, 1999.

[15] Yin-Wen Chang, Cho-Jui Hsieh, Kai-Wei Chang, Michael Ringgaard, and Chih-Jen Lin. Training and testing low-degree polynomial data mappings via linear svm. *The Journal of Machine Learning Research*, 99:1471–1490, 2010.

[16] Cyc. Graphic showing the maximum separating hyperplane and the margin. `http://en.wikipedia.org/wiki/File:Svm_max_sep_hyperplane_with_margin.png#filelinks`, February, 2008.

[17] John G Daugman. High confidence visual recognition of persons by a test of statistical independence. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(11):1148–1161, 1993.

[18] Robert DeLisle. Kohonen's self organizing feature maps. `http://www.ai-junkie.com/ann/som/som1.html`.

[19] Konstantinos I Diamantaras and Sun Y Kung. *Principal component neural networks*. Wiley, 1996.

[20] Thomas G. Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *arXiv preprint cs/9501101*, 1995.

[21] Sr. Dr. Nicholas M. Short. Graphic representation of hyperspectral data. `http://landsat.gsfc.nasa.gov/education/tutorials.html`, June, 2007.

[22] Qian Du and James E Fowler. Hyperspectral image compression using jpeg2000 and principal component analysis. *Geoscience and Remote Sensing Letters, IEEE*, 4(2):201–205, 2007.

[23] Rasmus EjrnÆs, Erik Aude, Bettina Nygaard, and Bernd Münier. Prediction of habitat quality using ordination and neural networks. *Ecological Applications*, 12(4):1180–1187, 2002.

[24] Anthony M Filippi and Rick Archibald. Support vector machine-based endmember extraction. *Geoscience and Remote Sensing, IEEE Transactions on*, 47(3):771–791, 2009.

[25] GM Foody. Approaches for the production and evaluation of fuzzy land cover classifications from remotely-sensed data. *International Journal of Remote Sensing*, 17(7):1317–1340, 1996.

[26] Alex S Fraser. Simulation of genetic systems by automatic digital computers vi. epistasis. *Australian Journal of Biological Sciences*, 13(2):150–162, 1960.

[27] M. Garcin. Essentials of hyperspectral imaging spectroscopy. `http://www2.brgm.fr/mineo/final.htm`, January, 2003.

[28] Noureddine Ghoggali, Farid Melgani, and Yakoub Bazi. A multiobjective genetic svm approach for classification problems with limited training samples. *Geoscience and Remote Sensing, IEEE Transactions on*, 47(6):1707–1718, 2009.

[29] Andrew A Green, Mark Berman, Paul Switzer, and Maurice D Craig. A transformation for ordering multispectral data in terms of image quality with implications for noise removal. *Geoscience and Remote Sensing, IEEE Transactions on*, 26(1):65–74, 1988.

[30] Robert O Green, Michael L Eastwood, Charles M Sarture, Thomas G Chrien, Mikael Aronsson, Bruce J Chippendale, Jessica A Faust, Betina E Pavri, Christopher J Chovit, Manuel Solis, et al. Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (aviris). *Remote Sensing of Environment*, 65(3):227–248, 1998.

[31] J Anthony Gualtieri and Robert F Cromp. Support vector machines for hyperspectral remote sensing classification. In *The 27th AIPR Workshop: Advances in Computer-Assisted Recognition*, pages 221–232. International Society for Optics and Photonics, 1999.

[32] Jean-François Guégan, Sovan Lek, and Thierry Oberdorff. Energy availability and habitat heterogeneity predict global riverine fish diversity. *Nature*, 391(6665):382–384, 1998.

[33] Tomoyuki Hamamura, Hiroyuki Mizutani, and Bunpei Irie. A multiclass classification method based on multiple pairwise classifiers. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition-Volume 2*, page 809. IEEE Computer Society, 2003.

[34] Joseph C Harsanyi and C-I Chang. Hyperspectral image classification and dimensionality reduction: an orthogonal subspace projection approach. *Geoscience and Remote Sensing, IEEE Transactions on*, 32(4):779–785, 1994.

[35] Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. *The annals of statistics*, 26(2):451–471, 1998.

[36] C Huang, LS Davis, and JRG Townshend. An assessment of support vector machines for land cover classification. *International Journal of Remote Sensing*, 23(4):725–749, 2002.

[37] Finn V Jensen. *An introduction to Bayesian networks*, volume 210. UCL press London, 1996.

[38] John R Jensen. *Remote Sensing of the Environment: An Earth Resource Perspective 2/e*. Pearson Education, India, 2009.

[39] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2005.

[40] Kwang In Kim, Keechul Jung, and Hang Joon Kim. Face recognition using kernel principal component analysis. *Signal Processing Letters, IEEE*, 9(2):40–42, 2002.

[41] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1):59–69, 1982.

[42] Teuvo Kohonen. *Self-organizing maps*, volume 30. Springer, 2001.

[43] John R Koza. Genetic programming as a means for programming computers by natural selection. *Statistics and Computing*, 4(2):87–112, 1994.

[44] Harold W Kuhn and Albert W Tucker. Nonlinear programming. In *Proceedings of the second Berkeley symposium on mathematical statistics and probability*, volume 5. California, 1951.

[45] Raymond Laë, Sovan Lek, and Jacques Moreau. Predicting fish yield of african lakes using neural networks. *Ecological modelling*, 120(2):325–335, 1999.

[46] James B Lee, A Stephen Woodyatt, and Mark Berman. Enhancement of high spectral resolution remote-sensing data by a noise-adjusted principal components transform. *Geoscience and Remote Sensing, IEEE Transactions on*, 28(3):295–304, 1990.

[47] Sovan Lek and Jean-François Guégan. Artificial neural networks as a tool in ecological modelling, an introduction. *Ecological modelling*, 120(2):65–73, 1999.

[48] M Lennon and G Mercier. Noise-adjusted non orthogonal linear projections for hyperspectral data analysis. In *Geoscience and Remote Sensing Symposium, 2003. IGARSS'03. Proceedings. 2003 IEEE International*, volume 6, pages 3760–3762. IEEE, 2003.

[49] M Lennon, G Mercier, and L Hubert-Moy. Classification of hyperspectral images with nonlinear filtering and support vector machines. In *Geoscience and Remote Sensing Symposium, 2002. IGARSS'02. 2002 IEEE International*, volume 3, pages 1670–1672. IEEE, 2002.

[50] Thomas M Lillesand, Ralph W Kiefer, Jonathan W Chipman, et al. *Remote sensing and image interpretation.* Number Ed. 5. John Wiley & Sons Ltd, 2004.

[51] Chuangmin Liu, LIANJUN Zhang, Craig J Davis, Dale S Solomon, Thomas B Brann, and Lawrence E Caldwell. Comparison of neural networks and statistical methods in classification of ecological habitats using fia data. *Forest Science*, 49(4):619–631, 2003.

[52] Wolfgang Maass. On the computational power of winner-take-all. *Neural Computation*, 12(11):2519–2535, 2000.

[53] Dimitris Manolakis, David Marden, and Gary A Shaw. Hyperspectral image processing for automatic target detection applications. *Lincoln Laboratory Journal*, 14(1):79–116, 2003.

[54] Fauvel Mathieu, Chanussot Jocelyn, Benediktsson Jón Atli, et al. Kernel principal component analysis for the classification of hyperspectral remote sensing data over urban areas. *EURASIP Journal on Advances in Signal Processing*, 2009.

[55] MATLAB. *version 7.12 (R2011a)*. The MathWorks Inc., Natick, Massachusetts, 2011.

[56] Farid Melgani and Lorenzo Bruzzone. Support vector machines for classification of hyperspectral remote-sensing images. In *Geoscience and Remote Sensing Symposium, 2002. IGARSS'02. 2002 IEEE International*, volume 1, pages 506–508. IEEE, 2002.

[57] Grégoire Mercier and Marc Lennon. Support vector machines for hyperspectral image classification with spectral-based kernels. In *Geoscience and Remote Sensing Symposium, 2003. IGARSS'03. Proceedings. 2003 IEEE International*, volume 1, pages 288–290. IEEE, 2003.

[58] Sebastian Mika, Bernhard Schölkopf, Alex Smola, Klaus-Robert Müller, Matthias Scholz, and Gunnar Rätsch. Kernel pca and de-noising in feature spaces. *Advances in neural information processing systems*, 11(1):536–542, 1999.

[59] Jonathan Milgram, Mohamed Cheriet, Robert Sabourin, et al. one against one or one against all: Which one is better for handwriting recognition with svms. In *Tenth International Workshop on Frontiers in Handwriting Recognition*, 2006.

[60] Tom M Mitchell. *Machine learning*. McGraw-Hill, Boston, 1997.

[61] Robert A Neville, C Nadeau, John Levesque, Tomas Szeredi, Karl Staenz, P Hauff, and Gary A Borstad. Hyperspectral imagery for mineral exploration: comparison of data from two airborne sensors. In *SPIE's International Symposium on Optical Science, Engineering, and Instrumentation*, pages 74–83. International Society for Optics and Photonics, 1998.

[62] Hisham Othman and Shen-En Qian. Noise reduction of hyperspectral imagery using hybrid spatial-spectral derivative-domain wavelet shrinkage. *Geoscience and Remote Sensing, IEEE Transactions on*, 44(2):397–408, 2006.

[63] M Pal and PM Mather. Support vector machines for classification in remote sensing. *International Journal of Remote Sensing*, 26(5):1007–1011, 2005.

[64] JD Paola and RA Schowengerdt. A review and analysis of backpropagation neural networks for classification of remotely-sensed multi-spectral imagery. *International Journal of remote sensing*, 16(16):3033–3058, 1995.

[65] Harris J Peshko, Paul Budkewitsch, M R St-Onge, R McGroger, R Hitchcock, and D Desnoyers. Hyperspectral units, livingstone river, baffin island, nuanavut. *Oper File 5054*, 2008.

[66] Antonio Plaza, Jon Atli Benediktsson, Joseph W Boardman, Jason Brazile, Lorenzo Bruzzone, Gustavo Camps-Valls, Jocelyn Chanussot, Mathieu Fauvel, Paolo Gamba, Anthony Gualtieri, et al. Recent advances in techniques for hyperspectral image processing. *Remote Sensing of Environment*, 113:S110–S122, 2009.

[67] RG Resmini, ME Kappus, WS Aldrich, JC Harsanyi, and M Anderson. Mineral mapping with hyperspectral digital imagery collection experiment (hydice) sensor data at cuprite, nevada, usa. *International Journal of Remote Sensing*, 18(7):1553–1570, 1997.

[68] Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *The Journal of Machine Learning Research*, 5:101–141, 2004.

[69] BD Ripley. Neural networks and pattern recognition. *Cambridge University*, 1996.

[70] Brian J Ross, Anthony G Gualtieri, Frank Fueten, and Paul Budkewitsch. Hyperspectral image analysis using genetic programming. *Applied Soft Computing*, 5(2):147–156, 2005.

[71] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 1:213, 2002.

[72] Floyd F Sabins. Remote sensing for mineral exploration. *Ore Geology Reviews*, 14(3):157–183, 1999.

[73] S Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *Systems, Man and Cybernetics, IEEE Transactions on*, 21(3):660–674, 1991.

[74] John R Schott. *Remote sensing: the image chain approach. 2*. Oxford University Press, 2007.

[75] M Suganthy and P Ramamoorthy. Principal component analysis based feature extraction, morphological edge detection and localization for fast iris recognition. *Journal of Computer Science*, 8(9):1428.

[76] Jack V Tu. Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. *Journal of clinical epidemiology*, 49(11):1225–1231, 1996.

[77] Vladimir Vapnik. *Statistical learning theory*. Wiley, 1998.

[78] Vladimir Vapnik, Steven E Golowich, and Alex Smola. Support vector method for function approximation, regression estimation, and signal processing. *Advances in neural information processing systems*, pages 281–287, 1997.

[79] Vladimir Naumovich Vapnik and Samuel Kotz. *Estimation of dependences based on empirical data*, volume 41. Springer-Verlag, New York, 1982.

[80] IB Vapnyarskii. Lagrange multipliers. *Encyclopaedia of Mathematics. Springer, Heidelberg*, 2001.

[81] Björn Waske and Jon Atli Benediktsson. Fusion of support vector machines for classification of multisensor data. *Geoscience and Remote Sensing, IEEE Transactions on*, 45(12):3858–3866, 2007.

[82] Qiang Wu and Ding-Xuan Zhou. Svm soft margin classifiers: linear programming versus quadratic programming. *Neural computation*, 17(5):1160–1187, 2005.